Université du Havre Laboratoire de Mathématiques Appliquées du Havre Ecole Doctorale SPMII

THÈSE

Présentée pour l'obtention de titre de Docteur en

Mathématiques Appliquées et Informatique

Optimisation des systèmes de stockage de conteneurs dans les terminaux maritimes automatisés

Hamdi DKHIL

Soutenue publiquement le 05 Octobre 2015 devant un jury composé par :

Professeur Cyril FONLUPT	Université du Littoral Côte d'Opale, Rapporteur
Professeur Abderrafiaa KOUKAM	Université de Technologie de Belfort-Montbéliard, Rapporteur
Professeur Saïd HANAFI	Université de Valenciennes et de Hainaut-Cambrésis, Rapporteur
Professeur Henri BASSON	Université du Littoral Côte d'Opale, Examinateur
Professeur Abdelkader SBIHI	Ecole de Management de Normandie, Examinateur
Professeur Adnan YASSINE	Université du Havre, Directeur de thèse, Examinateur

À l'âme de ma grande mère, Farida KAFFALA.

Remerciements

Cette thèse a été préparée au sein du Laboratoire de Mathématiques Appliquées du Havre, sous la direction du Professeur Adnan YASSINE, à l'université du Havre.

Je remercie en premier lieu Monsieur le Professeur Adnan YASSINE, mon directeur de thèse, pour ses aides multiples, ses encouragements et pour le temps précieux qu'il m'a accordé à chaque fois que j'en avais besoin.

Je tiens aussi à remercier Monsieur Aziz ALAOUI, Directeur du Laboratoire de Mathématiques Appliquées du Havre, ainsi que tous les membres du Laboratoire.

J'adresse mes vifs remerciements à Monsieur Cyril FONLUPT, Professeur à l'université du Littoral Côte d'Opale, ainsi qu'à Monsieur Abderrafiaa KOUKAM, Professeur à l'université de Technologie de Belfort-Montbéliard, pour avoir accepté de juger ce travail et d'en être rapporteurs.

J'adresse également mes sincères remerciements à Messieurs Saïd HANAFI Professeur à l'université de Valenciennes et de Hainaut-Cambrésis, Henri BASSON, Professeur à l'université du Littoral Côte d'Opale et Abdelkader SBIHI, Professeur à l'Ecole de Management de Normandie pour avoir accepté de siéger au sein du jury.

Je remercie particulièrement mes chers amis, Docteur Ihsen FARAH, Docteur Ezzeddine BENAISSA, Docteur Mohamed AMMAMI, Docteur Riadh MOUSSI, Mlle. Sara TFAILI, Mlle Mira AL KHARBOUTLY et M. Mohamad KHORBATLY pour leur aide précieuse durant ces années de recherche.

Je remercie infiniment mes très chers amis Abderrazzek BARRAH, Nabil SALMAN et Hussein SALMAN pour leur énorme soutien et pour leurs encouragements.

Je ne remercierais jamais assez certaines personnes, mais je vais essayer: je remercie infiniment ma mère, mon père, ma sœur et mon frère pour leur soutien constant, leur encouragement, leurs sentiments et leurs prières.

Introduction générale

L’ensemble des terminaux à conteneurs représente un nœud très important dans la chaîne internationale de transport de marchandises et agit directement sur l’économie mondiale. Vue l’évolution rapide du commerce mondial, la productivité des terminaux est mise à l’épreuve et l’optimisation des temps de stockage et de transfert de conteneurs, de l’espace de stockage et des coûts de stockage en nombre d’équipements devient un besoin primordial.

L’optimisation de stockage de conteneurs dans un terminal portuaire est un problème logistique très important qui a attiré l’attention des chercheurs depuis plusieurs décennies. Deux grands axes d’optimisation de stockage sont généralement étudiés : l’optimisation du temps de stockage et l’optimisation de l’espace de stockage. Ces deux problèmes sont souvent traités séparément. Quelques travaux traitent de la minimisation de la flotte de véhicules dans un terminal à conteneurs. La minimisation du nombre de véhicules utilisés dans un terminal à conteneurs et la minimisation du temps de déplacements des cavaliers sont les objectifs d’ordonnancement de tâches attribuées aux cavaliers.

Les systèmes de véhicules à guidage automatique (AGVS) ont connu ces dernières années une évolution sans précédente poussant les décideurs à mettre en œuvre des processus assez pointus permettant d’une part d’optimiser leur rentabilité, et d’autre part, de respecter des contraintes de plus en plus nombreuses.

De nombreuses analyses et études approfondies ont traité l’optimisation des AGVS. L'étude d’un AGVS commence avant même sa conception. En effet, des simulations assez pointues permettent de vérifier les trajectoires et le nombre de tâches à exécuter ainsi que le nombre de véhicules nécessaires pour réaliser le travail dans les meilleurs délais fixés par les décideurs. Vue la multitude des fonctions et des types de véhicules automatiques (transport d'engins lourds, transport de box à rouleaux et/ou de caisses-palettes de stockage, transport des palettes en fin de ligne, transport de produits alimentaires, transport de conteneurs, etc.) des études appropriées ont traité chaque cas en s’adaptant à sa particularité et son contexte. Les résultats généraux ou spécifiques des études des systèmes de véhicules à guidage automatique restent une référence

dans l'étude de chaque cas particulier.

Dans cette thèse nous nous sommes intéressés à un cas très particulier d'AGVS, il s'agit des terminaux à conteneurs automatisés, qui en plus des véhicules autoguidés, sont souvent équipés de grues de stockage automatiques (grues de cour), ce qui pousse souvent les scientifiques à considérer les problèmes d'ordonnancement intégré dans les terminaux automatisés ou semi-automatisés.

Nous traitons dans ce travail l'optimisation de plusieurs objectifs pour stocker les conteneurs avec une stratégie efficace et productive. Nous étudions le problème d'ordonnancement intégré considérant les trois équipements d'un terminal à conteneurs automatisé à savoir : les véhicules autoguidés, les grues à quai et les grues de baie. Nous étudions aussi le problème d'allocation d'emplacements de stockage aux conteneurs, à l'import, en intégration avec le problème d'ordonnancement des cavaliers. L'objectif principal de ce travail est la minimisation du coût opérationnel de stockage. Tous nos résultats pour les terminaux automatisés sont adaptés à plusieurs cas de terminaux non automatisés et de terminaux hybrides.

Nous abordons deux aspects d'optimisation des systèmes de stockages dans les terminaux automatisés, l'optimisation mono-objective du temps de manutention des conteneurs et l'optimisation multi-objective du coût opérationnel global.

Ce qui suit n'est pas une étude comparative des différents terminaux maritimes automatisés ni de l'impact du type du terminal sur sa productivité mais une proposition de solutions réalistes et efficaces adaptées à la nature de l'équipement utilisé et aux choix de routage des véhicules autoguidés. Nous qualifions les solutions que nous proposons de réalistes vu les nombreuses contraintes prises en compte dans nos modélisations ainsi que dans nos algorithmes de résolution, et nous les qualifions d'efficaces vu les avantages qu'offrent les résultats théoriques que nous avons développé ou présenté en termes de facilitation de résolution pour le problème d'ordonnancement de routage de véhicules et d'évaluation numérique et graphique de la qualité des décisions proposées pour le problème multi-objectif d'ordonnancement de véhicules autoguidés et d'allocation de conteneurs.

En effet, les principaux objectifs de cette étude sont: l'évaluation de coûts opérationnels

réalistes, le choix de routages avantageux pour des considérations pratiques et théoriques, l'évaluation de bornes inférieures pour la qualification des solutions proposées par l'optimisation multi-objective, le développement d'algorithmes adaptés à la complexité des problèmes considérés et la proposition d'outils efficaces pour la prise d'une décision suffisamment justifiée pour l'optimisation multi-objective.

D'ailleurs, ce travail représente un effort ambitieux pour la réalisation prochaine d'un outil logistique basé sur l'optimisation mathématique et la programmation informatique afin d'offrir une aide à la décision efficace et réaliste dans les terminaux maritimes automatisés.

Contents

Figures

Tables

Introduction

The main objective of this thesis is to build modeling and optimization efficient approaches dedicated to Automated Handling System (AHS) in Maritime Container Terminals (MCT) considering particularly the case of import. Many studies have been devoted to such a problematic, however, optimization of Maritime Automated Container Terminals (MACT) needs more global view and realistic approaches considering the importance of the relationship between mathematical results, computing processes and the realistic needs of the operators at MACT. In relieved state of art, despite the theoretical efficiency of many studies, applying their results needs the consideration of additional realistic constraints, the evaluation of economic operational cost with multi-criteria aspect and appropriate tools to facilitate the operator decision.

In our work, we develop modeling, mathematical and computing tools for MACT. We study and propose different terminal layouts by considering the vehicle traffic. Moreover, we consider all MACT kinds: MACT with Automated Straddle Carriers (Auto-Strad), MACT with Automated Guided Vehicles (AGV) and MACT with Automated Lifting Vehicles (ALV). Two problems are considered here in: Vehicle Scheduling Problem and Location Assignment Problem. Firstly, we study the Vehicle Scheduling Problem in MACT. We consider locations to use for container storage initially known. Realistic modeling is developed, mathematical results of the state of art are considered and ILOG CPLEX JAVA code is implemented for efficient resolution of large instances of the problem. We consider then the Integrated Problem of Location Assignment and Vehicle Scheduling (IPLAVS) in MACT, at import. We study two variants of this problem: a mono-objective variant that is minimizing the operating time of handling system and a Multi-Objective variant that is minimizing minimizing the operating cost which we evaluate considering eight objectives. Different tools are proposed to help operator decision, in particular, 2D-projections of Approximated Pareto Front and different indicators of efficiency.

Automated Container Terminals in Supply Chain

Supply Chain Management deals with the organization of different activities such as sourcing and procurement, conversion, and all logistics management activities. Moreover, it concerns the manufacturing operations. Not only it drives the coordination of processes and activities with and

across marketing, sales, product design, finance and information technology, but also it includes the coordination with partners, which can be suppliers, intermediaries, third-party service providers and customers. The accompaniment of efficient transportation processes insures the realization of world-class operations at the point of supply, production, and customer locations. Maritime transportation has been a catalyst of economic development and has provided the main vehicle for imports and exports. In Europe, almost 90% of the EU external freight trade is seaborne, whereas 40% of the intra-EU exchanges is represented by short sea shipping.

On August 15th, 1962, the Port Authority of New York and New Jersey opened the world's first container port, Elizabeth Marine Terminal. The concept had been developed by the McLean Trucking Company in 1956. Container terminals represent an essential element in the supply chain management where cargo containers are transshipped between different vehicles, for onward transportation. A container terminal is said to be a maritime container terminal if the transshipment occurs between container ships and land vehicles such as trains or trucks. The main maritime container terminals are located around major harbors.

New generation of maritime container terminals using high technology for the handling operations represents a remarkable progress in the field of container transportation system. Such a generation is said to be automated container terminal. The competition between major international ports enriches the choice of automation, which is also explained by its guaranteed security, ease of organization and traceability and higher productivity due to 24 work hours per day. We can distinguish the following automated container terminals, CTA in the port of Hamburg (Germany), ECT in Rotterdam (Netherlands), Automated Container Terminal of Ottawa (Canada), Brisbane Container Terminals and Sydney International Container Terminal (Australia).

The benefits for terminal operators are essentially:

- ✓ An enormous increase in handling performance and improvements in terminal performance
- ✓ A reduction of wage cost
- ✓ An improved utilization of existing stack areas
- ✓ An increased productivity and a competitive cost reduction per move

- ✓ Automated vehicles and automated stacking cranes efficient work together.

In recent years, methodological progress regarding container terminal operations have considerably bean improved. However, mathematical optimization with more global point of views and multi-criteria objective are rare in the literature. In our study, we consider two optimization problems in automated container terminals at import; the first is the vehicle scheduling problem; and the second is the integrated problem of location assignment and vehicle scheduling. In the first part of our study, we propose different traffic layout adapted to the two studied problems and to every kind of automated container terminal. We also introduce relevant literature reviews studying the optimization of container handling systems at maritime terminal, the optimization of general automated guided vehicle system and the multi-objective optimization in general, and finally, in a particular context of maritime container terminals. In the second part, we solve the planning of QC-AV-ASC (Quay Cranes-Automated Vehicles - Automated Stacking Cranes). We present an effective model for every kind of traffic layout. Moreover, we propose an efficient bi-objective model which is important to determine the optimal storage time and the minimal number of required AVs. CPLEX resolutions are used to prove the efficiency of our modeling approach.

In the third part of this thesis, we explore a problem which has not been studied yet: the integrated problem of location assignment and vehicle scheduling (IPLAVS), in Maritime Automated Container Terminal (MACT) at import. This part represents a new and realistic approach of MACT optimization considering both mono-objective and multi-objective variants.

In this thesis, the multi-objective integrated problem of location assignment and vehicle scheduling is studied for the first time considering all the state of art.

Chapter 1

Maritime Automated Container Terminals

Problematics, equipment and traffic layout

1.1 Introduction

Maritime Container Terminals (MCT) plays a crucial role in global logistic networks. Because of the ever-increasing quantity of cargo, terminal operators need solutions for different decisional problems. In the maritime terminal, at the boat arrival or departure, we observe five main problems: the assignment of berths, the assignment of query cranes, the allocation of storage space, the optimization of stacking cranes work load and the scheduling and routing of vehicles. A good cooperation between the different equipments in the terminal is important in order to optimize the productivity of Container Handling System (CHS). In an automated container terminal, numerical solutions have become essential to optimize the operators decisions. Many recent researches have discussed the optimization of equipment scheduling in Maritime Automated Container Terminal (MACT).

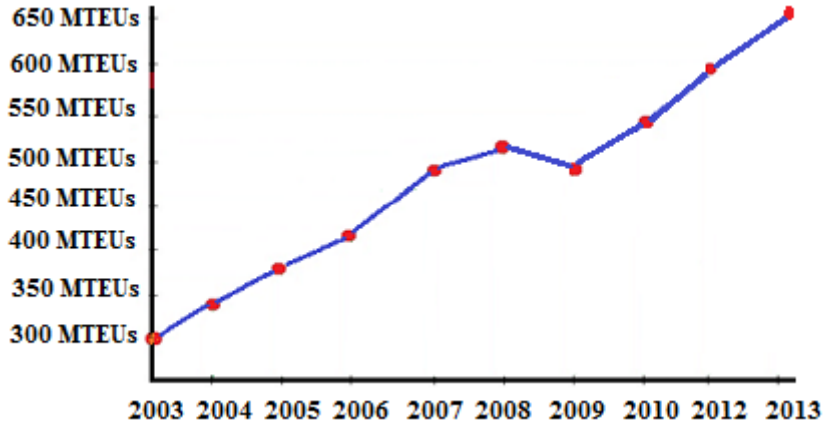
We identify three kinds of MACT, considering their equipment; MACT using Automated Guided Vehicles (AGVs), Automated Stacking Cranes (ASCs) and Quay Cranes (QCs), MACT using ALVs, ASCs and QCs and MACT using Auto-Straddle-Carriers and QCs (without ASCs). In our study, we consider these three cases in Maritime Automated Container Terminal (MACT) at import.

1.1.1 Word container traffic evolution

In this part, we give some statistics about word container traffic evolution in the last years to understand why terminal operators need more and more efficiency for their handling tasks. We consider the data bases of World Bank and French Center of Maritime Studies (French Ministry of Development and Transport).

The world container port traffic grew from 300 million TEUs (Twenty Equivalent Units) in 2003 to more than 601 million TEUs in 2012 and the world container fleet capacity grew from 1.7 million TEUs in 1990 to 16 million TEUs in 2008. If we compare the number of orders for container ships to the number of container ship deliveries in 2007 (see Table 1), we can conclude that the world container traffic requires more and more ships and especially ships with a capacity of more than 10 000 TEUs. Notice that in 2007, 134 of these container ships were recorded in order books and only 7 were delivered. In the same year, the orders of container ships that can carry more than 7500 TEUs represented 34 % of the container ships ordered.

World container port traffic between 2003 and 2013



M TEUs: Million Twenty Foots Equivalent Units

Figure 1

Data source: www.worldbank.com

Considering this important evolution of the world container traffic, maritime terminals need new organizational strategies in order to insure more efficiency for their Container Handling Systems (CHS). Many analyses and mathematical works treat the optimization of handling operations in maritime container terminals. However, the multi-objective aspect of CHS optimization is not sufficiently considered in these works.

Container-Ships commands and deliveries in 2007

ship capacity(TEUs)	Commands Number of ships / total capacity (TEUs)	Deliveries Number of ships / total capacity
>10 000	134/ 1 659 092	7 / 96 124
7500 / 10 000	78 / 673 778	34 / 300 516
6 000/7500	39 / 257 014	27 /181 630
5250 / 6000	9 / 49 950	5 / 29 112
4000 / 5250	130 / 576 015	65 / 305 169
3 000 / 3 999	31 / 108 374	25 / 88 670
2 000 / 3 000	63 / 160 465	43 / 113 481
1 000 / 2 000	126 / 177 116	115 / 161 241
<1000	62 / 51 359	13 / 11 732
All ship	606 / 3 637 957	400 / 1 362 881

Table 1

Data source: French Center of Maritime Studies, Ministry of Development and Transport

In next table we present some statistics about the Evolution of World Container-Fleet Capacity (WCFC) between 1990 and 2011.

Evolution of World Container-Fleet Capacity (WCFC)

Year	WCFC	Number of Container Ships	Average ship capacity
1990	**1.7 Million TEUs**	**1236**	**1389 TEUs**
2000	**4.5 Million TEUs**	**2611**	**1733 TEUs**
2008	**10.9 Million TEUs**	**4318**	**2530 TEUs**
2011	**16 Millions TEUs**	**5537**	**2897 TEUs**

Table 2

Data source: French Center of Maritime Studies,

Ministry of Development and Transport

1.1.2 Handling tasks at Maritime Container Terminal

At import, MCT guarantees especially three tasks: unloading containers from ships, stacking containers in storage space and finally delivering containers to shippers and consignees. At export, MCT guarantees the same tasks but in the opposite order. These three tasks are composed by other tasks which represents important optimization problems: berth allocation, yard planning, stowage planning, quay crane scheduling, vehicle scheduling and routing (straddle carriers, Automated Guided Vehicle, Automated Lifting Vehicle etc.), yard crane scheduling, logistics planning of operations. Logistic planning provides an efficient coordination between the different equipments and decisions at MCT.

Many kinds of handling systems are used at maritime container terminal, in this thesis we study the case of CHS (Container Handling System) using Automated Vehicles (AVs) to transfer containers between the quay and the storage space. We identify three kind of AV used at Maritime Automated Container Terminals: Automated Guided Vehicle (AGV), Automated Lifting Vehicle (ALV) and Auto-Straddle-Carrier (Auto-Strad). Different blocks

compose the storage space, every block is a set of storage bays and each bay is composed by different storage locations (figure 2). Using Auto-Strad in MCT, there's no stacking crane, the vehicles (Auto-Strads) directly access to the storage bays. If the terminal is equipped with AGVs or ALVs, the transfer of container in storage bay is insured by Automated Stacking Cranes (ASCs).

In an automated container terminal (ACT), the time of handling operations depends on the interactions between the different storage equipments. Different researches are established to improve the handling systems performance.

Storage block in MACT with AGVs or ALVs

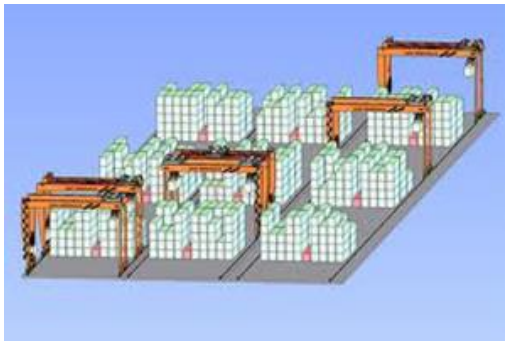


Figure 2

Storage block in MACT with Auto-Strads

Figure 3

The problem of AGV scheduling was treated in the general context of AGVS and in the particular context of ACT. AGVS is a materials handling system that uses automated vehicles which are programmed to achieve tasks between different manufacturing and warehouse stations. It represents a very important innovation in international transport and logistics. ACT is one of the most famous examples of AGVS. Studies of AGVS optimization have different objectives: maximizing the throughput, maximizing the vehicle utilization, minimizing the inventory level, minimizing the transportation costs, and maximizing the space utilization. Approaches that are used in AGVS optimization can be classified in two types: analytical approaches and simulation-based approaches. Analytical methods are mathematical techniques such as queuing theory, integer programming, heuristic algorithms, and Markov chains. A number of analytical approaches to AGVS optimization have been proposed in the literature.

In the next sections, we describe the most important studies treating optimization of handling system of MCT (Maritime Container Terminal).

1.2 Literature review

1.2.1 Problems of minimizing AGV fleet size in AGVS and ACT

Historically AGVS have not been produced in high volume. Then in AGVS determining the minimum number of required vehicles is crucial to improve the global system productivity.

Muller [2] used rough estimates of total AGV travel times and transport frequency to solve the AGV system case. Maxwell and Muckstadt [3] discussed the deterministic case of the problem. They considered the random aspect of the problem: variation of arrival pattern of jobs and vehicles speed and they developed an integer programming model to minimize the number of required AGVs. In the study of Rajota et al., [4] other parameters are considered: load handling times, empty travel time… The authors developed a mixed integer programming model to solve the problem. Sinriech and Tanchoco [5] have developed a multi-objective model which keep the total cost of AGV system down and increases the system utilization. The problem is treated by I FA Vis [6], in the ACT context he developed new planning concepts to minimize the AGV fleet size and he applied it to the container terminal case considering a deterministic model with defined time windows for each container load. He proposed two methods to solve the problem: an integer programming model and a formulation of the problem as a set of partitioning sub problem.

1.2.2 Minimizing vehicle fleet size in other contexts

Two similar problems are discussed in the literature. The first problem is to determine the minimum number of operators required to accomplish a known schedule of tasks. This problem was treated by Phillips and Garcia-Diaz [7]. They used a bipartite network where the maximum flow indicates pairs of tasks assigned to the same operator. Then they proposed to determine the arcs of the maximum flow to obtain the list of tasks for each operator. Ford and Fulkurson [8] discussed this problem and used a partial order of tasks: tasks i precedes task j if the start time of i is earlier than the start time of j and if the two tasks can be achieved by the same operator. They solved the problem with the determination of minimum chain decomposition. The second analog problem is the tanker scheduling. Dantizig and Fulkerson [9] described a deterministic model to solve the tanker scheduling problem with linear programming formulation and simplex algorithm. Ahuja et

al., [10] proposed another approach to solve the same problem: they introduced a minimum cost flow formulation of the problem and used a minimum cost flow algorithm to minimize the fleet size of the main problem.

1.2.3 Minimizing the makespan in AGVS and ACT

The problem of minimizing the makespan is treated in the general AGVS context. In 1984 Ebeglu and Tanchoco [11] developed a dispatching rules method for AGVs scheduling. Tanchoco et al., [12] discussed real-time control strategies for multiple-load AGVs. The models and methods applied to AGVS seem to be generally applicable and need to be adjusted for more specific contexts. Research works of minimizing makespan in ACT are recent, especially with the integrated aspect of QC-AGV-ASC problem (AGV or ALV). Chen et al., [13] treated the scheduling of AGVs. They developed a dispatching approach and simplified the QC task considering it available to AGV loading or unloading which cannot ensure the solution optimality for the multiples QCs case. Kim and Bae[14] developed a model with fixed pick up time for each container and they proposed heuristic solution for more general cases. To our knowledge Meersman [1] was the first researcher to consider the integrated QCs, AGV and ASC scheduling problem. He showed that this problem is NP-Hard and developed theoretical results for the problem of scheduling ASC-AGV-QC tasks. He studied static traffic layout (layout with one fixed path for the set of tasks) and multiple paths traffic layout (layout with different possible paths for the set of tasks). Meersman used branch and bound and beam search algorithms to solve the problem using theoretical results to establish valid inequalities. Bae et al., [15] developed a dynamic berth scheduling method for minimizing the cost of the vehicles travel during the loading or unloading of the ship. The approach takes into account many constraints and real dynamic situations.

1.2.4 Storage space optimization at maritime container terminals

In 1997, Kim [16] evaluated the number of re-handles in container yards. The author discussed a set of equations to estimate this number. In 1999, Kim and Kim [17] developed a beam search algorithm for the straddle carrier routing problem at export. The approach comprises the container location problem and the carrier routing problem. The authors treated only one objective,

minimizing the total travel distance of straddle carriers in the yard. In another paper, Kim and Kim [18] developed a segregating space allocation modeling for import container inventories in port container terminals. The objective is to minimize the expected total number of re-handles. The authors discussed different procedures to solve the problem. In 2000 Kim et al., [19] discussed driving decision rules to solve the storage space allocation problem. They considered the goal of minimizing the number of relocation movements expected for the loading operation. The authors developed a dynamic programming model. To solve the real time problem, they used a decision tree considering the optimal solutions of the dynamic programming model. In 2007, Chen et al., [20] presented a Tabu search algorithm to solve the integrated scheduling problem of container handling systems in a maritime terminal. The authors presented and discussed the problem as a hybrid flow-shop scheduling problem.

1.2.5 Multi-criteria AGVS scheduling models

With the increasing automation of manufacturing systems, the use of efficient and multi-criteria decision systems is very important to optimize productivity. AGV systems seem to be the most famous example. A good evaluation of AGVS cost must take into account different characteristics: vehicle dispatch, load and unload, central controller, complex host interface, product tracking, multiple path layout etc.

Dahlstrom and Maskin [21] and Muller [22] have addressed the economical aspects of AGVS; the two papers compared the cost of different material handling systems. Sinriech and Tanchoco [23] have developed a multi-objective model which keep the total cost of AGVS down and increases the system utilization. They assumed that the AGVS cost is a formulation of operating costs (maintenance, energy...) and design costs (vehicle supervisory controller, vehicles, batteries, chargers, communication links etc). Another case is studied by Vu D N and Kap H.K [24]; we can describe this case as a multiple fixed paths layout. Maxwell and Muckstadt [25] discussed the deterministic case of the problem. They considered the random aspect: variation of arrival pattern of jobs and vehicle speed. They developed an integer programming formulation to minimize the number of required AGVs. In Rajotia et al., [26] other parameters were considered: load handling times and empty travel time. Golias et al. [27] formulated and solved the discrete space and

dynamic vessels arrival time (DDBSP). The novelty was to consider the multi-criteria aspect of the problem. Two objectives are maximized: the customer satisfaction and the reliability of the berth schedule. Authors used a multi-objective genetic algorithm to solve the problem. Giallombardo et al., [28] studied the integrated problem of berth allocation and QC (Quay Crane) scheduling. Two objectives are considered, the first is to maximize the total value of chosen QC profiles and the second is to minimize the housekeeping costs of the transshipment flow. An economic analysis of the value of QC assignment profiles and of yard-related costs in a transshipment context is discussed. To our knowledge, Bish et al., [29] studied for the first time, the problem of location assignment and AGV (Automated Guided Vehicle) scheduling in automated container terminal. The authors proved the NP-Hardness of the integrated problem. The problem was studied as a mono-objective optimization problem with the objective of minimizing the handling time. The vehicle schedule and location assignment are optimized but the waiting-times in bay entry (AGV wait for stacking crane in bay entry) were not considered in this work.

The different approaches proposed analyze only limited parts of the MCT handling system and do not sufficiently cover the set of handling operations in the terminal. Some approaches consider a combination between two chronologically successive optimization problems in MCT, but a limited set of researches considers the multi-objective aspect of these integrated problems. The multi-objective approaches propose at most three-objective optimization models treated generally as a mono-objective problem using a linear function of the different studied objectives. The multi-objective problems at maritime terminal are, at the most of the time, non convex problems, then if we solve them using a linear function of the considered objectives some efficient solutions (non-dominated solutions) will never be proposed, even if we use a large number of linear objective function.

MOOP are not sufficiently studied in the general context of MCT and the particular context of container terminal managed by straddle carriers, especially if we consider the number of studied objectives and the approaches of resolution. In our study, we propose a multi-objective modeling and resolution approach with eight objectives. To solve the problem, we

developed a new Multi-Objective Tabu Search Algorithm (MOTSA) that we will develop later in this thesis.

1.2.6 Multi-objective optimization in general context

1.2.6.1 Continuous multi-objective optimization

As the name suggests, Multi-Objective Optimization (MOO) considers different goals in only one global problem. In industry and logistics, the first resolutions of Multi-Objective Optimization Problems (MOOP) have transformed these problems to Single-Objective Optimization Problems (SOOP) in order to solve them. However, there are many differences between these two cases. In fact, decision-making for MOOP needs a new generation of Multi-Objective Algorithms (MOA).

After resolution of MOOP, the result is generally a set of solutions and the operators have to choose one of them. To select one efficient solution, different methods are proposed. We describe these methods as Multi-Objective Election Methods (MOEM). The most used MOA in the literature are meta-heuristic algorithms. Genetic Algorithm (GA), Evolution Strategies (ES), Simulated Annealing (SA) and (TSA) are particularly used. The most used meta-heuristic for MOOP is Genetic Algorithm (GA).

Deb et al., [30] developed multi-objective GA named (NSGA-II). NSGA-II is a non-dominated sorting genetic multi-objective evolutionary algorithm. Authors compared NSGA-II to different effective variants of GA. NSGA-II performed the other algorithms for nine test problems. NSGA-II highlights three famous difficulties concerning multi-objective evolutionary approach: the $O(MN^3)$ computational complexity (where M is the number of objectives and N is the population size), the non-elitism and the determination of sharing parameters. In fact, the algorithm is a $O(MN^2)$ computational complexity. For each generation, the best N solutions from parent and offspring populations are selected, giving the approach an elitist factor. NSGA-II is a generic algorithm and can be implemented for different continuous or binary problems. Jaeggy et al. [31] developed MOTSA for continuous optimization problems. Inspired by path relinking strategies in discrete optimization, the authors developed a resolution approach. The objective was to keep the overall MOTSA computational cost at a minimum threshold.

Hansen [32] developed MOTSA using parallel searches. Each Tabu search algorithm uses variable objective weights and considers a total variable objective equal to a linear weighted sum of the multiple objectives. Each search (thread in practice) performs these weights during the run time with dynamic update. This strategy is effective if the Region of Pareto Front (PFR) is convex. Otherwise, some Pareto optimal solutions cannot be found by a weighted sum method. Jaeggy et al., [33] developed parallel MOTSA for continuous optimization problems. They compare MOTSA and NSGA-II (developed by Deb et al., [30]) to test parallel MOTSA efficiency. Considering the authors experiments, parallel MOTSA performs NSGA-II on five test functions out of nine. Jaeggy et al. [34] adapted MOTSA for real-world optimization problems considering its handling constraint.

1.2.6.2 Multi-objective combinatorial optimization

Considering Multi-Objective Combinatorial Problems, exact methods have very limited performance. MOSA (Multi-Objective Simulated Algorithm) is used in literature to solve different problems. The method is particularly used to solve assignment problems [35], production scheduling problems [36] and packing problems [37]. Gandibleux and Fréville [38] developed MOTSA (Muti-Objective Tabu Seach Algorithm) for combinatorial problems, they use dynamic weights updated at each iteration of the neighborhoods' exploration. The algorithm updates each weight proportionally to the deviation of associated objective. After every update, the current list of weight is put tabu. Hansen [39] developed MOTSA using distance between Pareto-optimal solutions to evaluate the update of every weight at each iteration of the algorithm.

1.3 Terminal equipment

If we consider container terminal equipment, and particularly the type of Automated Vehicles (AVs) used, we can identify clearly three kinds of MACT: MACT with Automated Guided Vehicles (AGVs), MACT with Automated Lifting Vehicles and MACT with Auto-Straddle-Carriers (Auto-Strads). Quay Cranes are common handling equipment in maritime container terminals.

Quay Cranes in MCT

Figure 4

Source: hugercrane.com

1.3.1 Maritime container terminals with AGVs

AGV is not able to load or unload containers. This particularity causes some waiting times at bay entry. In fact, when AGV is at the bay entry, the ASC unload the AGV and transfer container to its storage location. AGV can move to transfer next container when ASC picks up the container (unload AGV), but often, when an AGV x, is at bay entry, ASC is transferring some container and may be, the ASC has to serve others AGVs before serving x. For the same raisons waiting times are caused under QC, when AGV has to be served by QC, which has to unload a container from the ship and load it on the AGV.

AGVs waiting for ASC

Figure 5

Source: www.ebanataw.com

1.3.2 Maritime container terminals with ALVs

ALV is able to load and unload container, then it has not to wait for ASC to unload container or for QC to load it. When ALV is at bay entry, it unloads container and moves to load next container under QC. There is no waiting time under the cranes (ASCs and QCs).

Automated Lifting Vehicle (ALV) transferring container

Figure 6

Source: http://www.sectormaritimo.com

1.3.3 Maritime container terminals with Auto-Strads

Auto-Strad is able to load and unload container, but it enters to the bay, stores container in its exact location and then moves to pick-up next container to transfer under QC. When an Auto-Strad enters to a bay, the access to that bay will be blocked for the others Auto-Strads during a given security time. This security particularity causes waiting time at bay entry.

Auto-Strad Acceding to storage bay

Figure 7

Source: www.kalmar.com

1.4 Traffic layouts for the vehicle scheduling problem

Different automated container terminal layouts can be considered. Meersman [1] presents two possible architectures: a simple layout with static AGV traffic and a complex layout with multiple fixed paths and a common return point for all the AVs. Vu D. Nand Kap H.K [24] studied the second case also. Note that Maritime Automated Container Terminal (MACT) using Automated Guided Vehicles (AGVs) or Automated Lifting Vehicles (ALVs) as Automated Vehicles (AVs) use also Automated Stacking Cranes (ASCs), while, MACT using Auto-Straddle-Carriers (Auto-Strads) as AVs (Automated Vehicles) don't use ASCs because Auto-Strad enters the bay and stores container in its storage location. Note also that we use the notion of ASC Points for terminals using ASCs, and QC Points: ASC Points are the places where ASCs pick up containers and QC Points are the places where QCs unload containers from the ship and where AVs start the transfer of containers to ASC points (these notions will be used in the next parts). For the two first models, we consider also Point A as a final position in the path for every task. We consider next, three terminal layout possibilities.

1.4.1 One-path layout for AGVs

The model supposes static AGV traffic and does not take into account traffic security. We consider that all AGVs have the same path for each task. We can describe this case as a one-path layout. We consider the import case and the export case as symmetric and the scheduling problem is the same. Point A is the final point of every task. All AGVs have the same task path. We assume that the terminal's routes have one possible direction and that many AVs can use the same path at the same time without risks. The AGVs start under QC, then go to point B, then to the ASC point (where there is a possible waiting time for AGV) and finally they return to point A. Before starting its task, every AGV has to wait until the end of the last QC task.

With this model of terminal layout, the optimization can minimize only the sum of waiting times at the QC and ASC points, because routing path is initially known for each container. This layout is treated by Chang Ho Yang and all [40]. For the decision, only the vehicles schedule is to be identified because in each case we can choose the first AGV returning to Point A for the next task.

One-Path layout

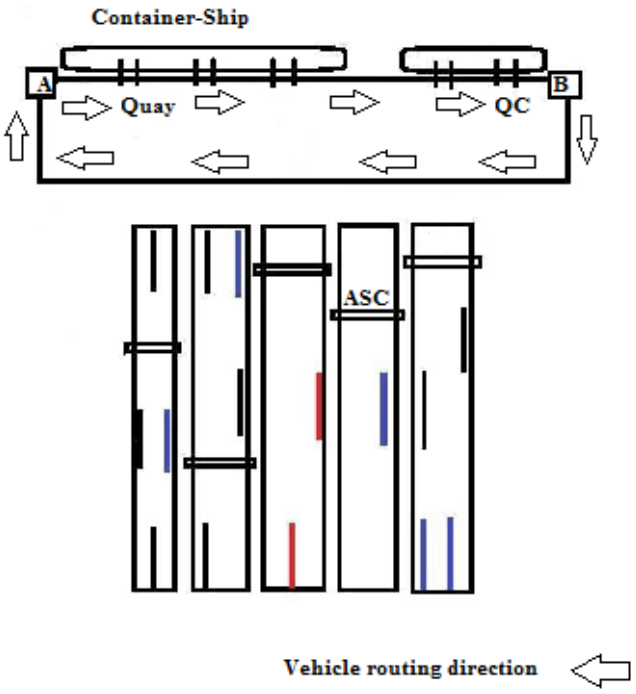


Figure 8

1.4.2 Multiple fixed paths layout for AGVs

Point A (see Fig.9) is the final point of every task. All AGVs have a known task path; they start under QC then choose the shortest path to the ASC Point, finally going to point A. The paths are not the same for all tasks but each path is initially known, they depend only on the ASC and QC positions. Before starting its tasks, every AGV has to wait enough time so that to not cause an accident with the predecessor AGV at QC buffer space.

Multiple fixed paths layout

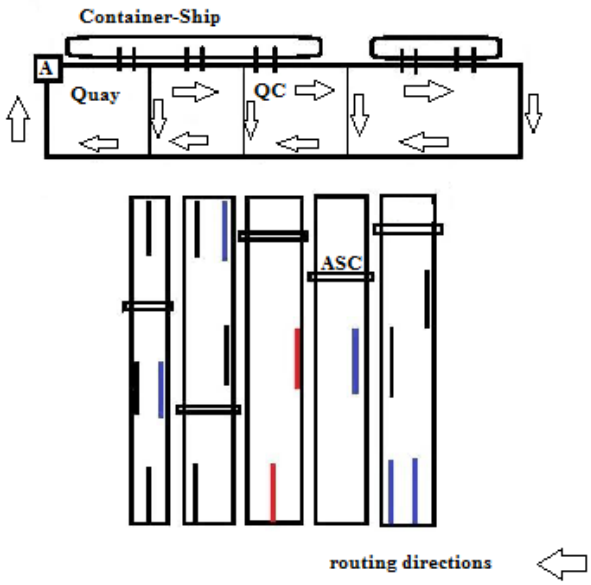


Figure 9

With this model of terminal architecture, we have to minimize only the sum of waiting times at the QC and ASC Points, because routing path is initially known for each container. We assume that the terminal's routes have one possible direction and that many AGVs can use the same path at the same time without any risk. For decision, only the vehicles schedule is to be identified because in each case, we can choose the first AGV returning to Point A for the next task. For the two first cases (one-path and multiple fixed paths layouts), we optimize the AGV scheduling problem with the same linear model.

1.4.3 Multiple variables paths layout for AGVs

This third case is the most complex architectural model. The travel times are unknowns because for each task, the AGV does not return to a common final point (Point A in FIG.2 and FIG.1) but moves directly to its next task. The travel time between the current task and the next one is unknown and depends on the choice of the next task.

Multiple variables paths layout

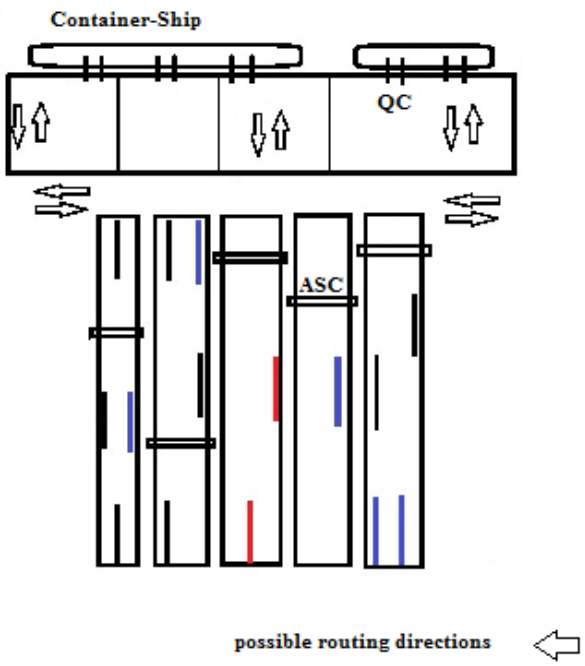


Figure 10

We assume that the terminal's routes have two possible directions and that many AGVs can use the same path at the same time without risks. In the static and the multiple fixed paths traffic models, when we optimize container handling time, we have only to identify container transfer schedule because in each case, we can choose the first AGV (returned to Point A) for the next task. In this traffic case, the choice of AGV for some tasks is important because AGVs do not terminate their tasks at the same point. Thus choosing the first free AGV for the next task is not a good idea: we have to release a double scheduling (Container, AGV).

1.4.4 Traffic layouts for ALVs

For MACT with ALVs, we consider the same particularities of traffic layouts as in part 1.4.1, 1.4.2 and 1.4.3, except the fact that ALV does not wait for ASC.

1.4.5 Traffic layouts for Auto-Strads

For MACT with Auto-Strads, we consider the same particularities of traffic layouts as in part 1.4.1 and 1.4.2, except the fact that Auto-Strad does not wait for ASC at bay entry, but it waits for the end of storage bay occupation by other vehicles, then enters the bay, stores the container and moves to the next position.

In the next two figures, we present two possible traffic layouts for MACT with Automated Straddle Carriers (Auto-Strad). In the first one, two common point of vehicle routing are considered (Point A and point B). In the second traffic layout, only one common point of vehicle routing is considered, that point is a return point for every vehicle, in other words, it represents the final point of each transfer task.

Traffic layout with two common point of vehicle routing

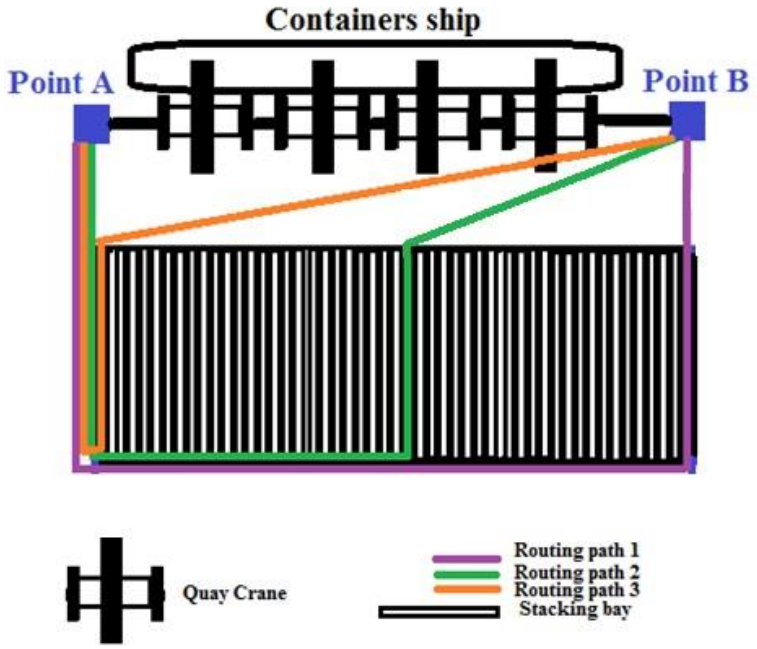


Figure 11

Traffic layout with on common point of vehicle routing

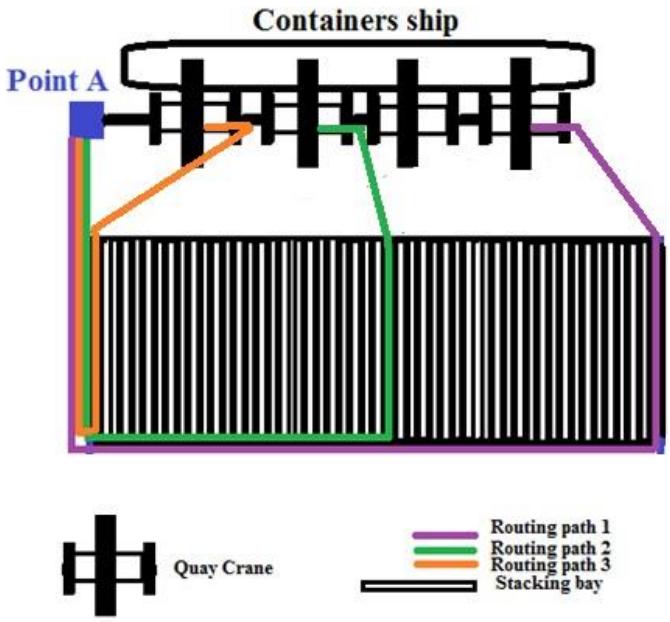


Figure 12

1.4.6 Traffic layouts for IPLAVS

1.4.6.1 The case of IPLAVS for AGVs

For a general traffic layout without the points A and B, as presented in figure 10, considering the Integrated Problem of Location Assignment and Vehicle Scheduling, we have to determine for every task the container to transfer, the vehicle to use and the exact location of storage. For a traffic layout with return point A and without the point B as presented in figure 9, we have to determine for every task the container to transfer and the storage location. The AGV to be used is the first vehicle arriving to point A (start point). For a simple traffic layout with the points A and B, as presented in figure 9, we have to determine for every task only the storage location. The AGV to be used is the first vehicle arriving to point A. The container to be transferred is one of the first

containers that were unloaded by QCs. This decisional particularity does not change the optimality of IPLAVS considering the minimization of operating time.

We propose a fourth traffic layout with two particularities; firstly, the points A and B as in static layout presented in figure 8 are considered; secondly, AGV take short cut path to move from storage bay to point A (common return point) and from point B to storage bay.

Considering the first particularity, to solve IPLAVS, we have only to determine for each task the storage location as for the static layout. The second particularity ensures more fluidity to vehicle traffic.

Proposed traffic layout

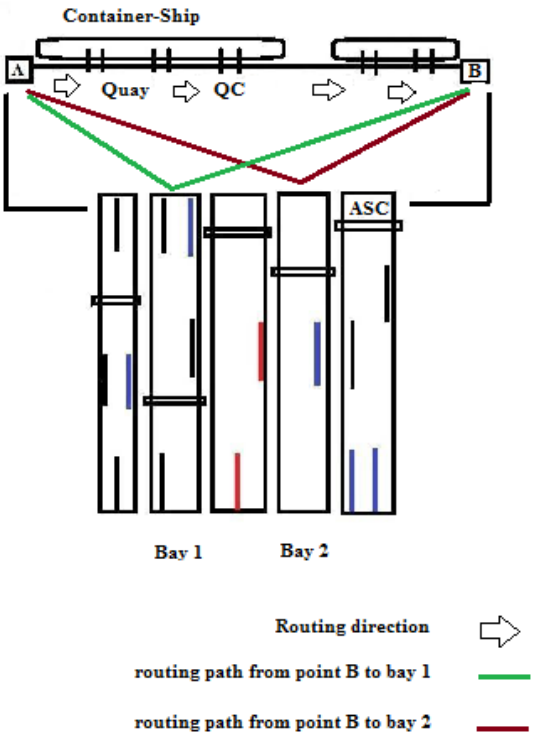


Figure 13

1.4.6.2 The case of IPLAVS for ALVs

For MACT using ALVs, we consider the same particularities as in part 1.4.4.1, excepting the fact that ALV does not wait for ASC.

1.4.6.3 The case of IPLAVS for Auto-Strads

For MACT using Auto-Strad, we consider next traffic layout.

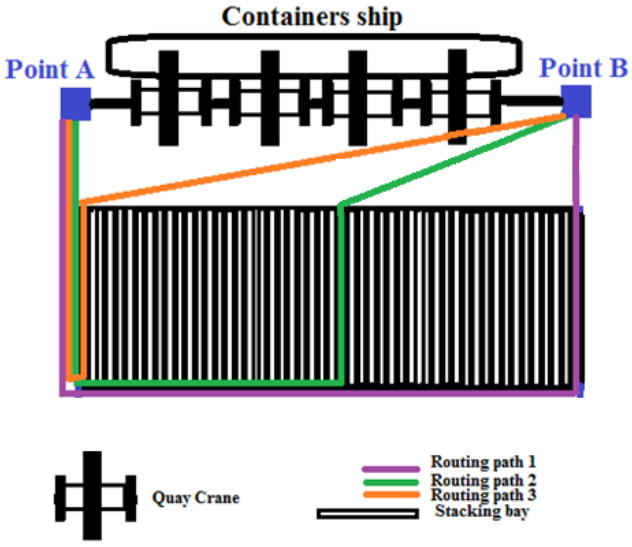


Figure 14

✓ Only one quay is considered. Then, for multiple quays terminal, we consider the problem for only the container ships allocated to the same quay.

✓ Considering QC (Quay Crane) scheduling, the container-unloading schedule is initially known for each QC. In fact, we have to determine only straddle carrier schedule and storage location assignment. Considering our layout choices, when storage location assignment is determined, the straddle carrier assignment and schedule are naturally identified (see the end of this part).

✓ Our modeling supports multiple container-ship unloading operations considering compatible arrival times. To support multiple-ship instances, it is crucial to know the exact

date of container ship arrivals, the number of QCs used, and the schedule of container handling for each QC.

✓ The vehicle routing in the quay has to respect a unique direction.

✓ Straddle carriers picks up containers under QCs respecting a known global schedule. This schedule is based on the sub-schedules of containers unloading from ships (QC schedule). If n is the number of QCs used, for every n successive transfer tasks, straddle carriers have to serve all QCs; in other words, QCs are served successively by the fleet of vehicles (note that vehicles are straddle carriers for all this paper). QCs operate loading tasks in parallel and do not wait for vehicles arrivals. Theoretically, this scheduling constraint influences the problem feasibility and optimality. However, numerical results show that it does not affect the problem for a general configuration of storage space.

✓ We consider that vehicles are initially in the preloading position near QCs.

✓ Every handling task begins when a straddle carrier picks up the container under QC.

✓ Every handling task ends when astraddle carrier stacks the container in the associated location.

✓ Considering the straddle carrier traffic, we define the quay entry (Point A in figure 14) and the quay exit (Point B in figure 14) as the entry and the output of the part of the quay reserved for QCs used.

✓ After picking up containers under quay cranes, the straddle carrier moves to quay output, then it moves to the entry of the bay where the storage location associated to the container is. It waits sufficiently to avoid an accident with the last vehicle entering the same bay and finally it transfers the container to its storage location in the bay. After the end of this handling task the vehicle moves to the quay entry and then it moves to the position of the next container picking up task under QC.

✓ At anytime of the process, it is easy to determine which straddle carrier to use for the next container-handling task. In fact we only have to choose the first vehicle returning to the

quay entry (Point A in figure 14).

✓ Considering the storage space, the set of free storage locations is initially known, however, we have to determine the storage locations to use for stacking containers. The free storage bays are naturally determined by the free storage locations and the storage bays to be used are determined by the storage locations to be used for each solution to the problem.

✓ With the chosen layout, each handling task is naturally identified by the associated container.

Research outputs

In the first part of this thesis, for each type of ACT, we propose mathematical models and an exact resolution of handling tasks planning, the problem of tasks in an automated container terminal. Our first objective is to minimize the makespan (the time when the last task is achieved). The second objective is to minimize the number of required automated vehicles. In the second part of the thesis, we propose new and efficient Mono-Objective and Multi-Objective Optimization approaches (M.O.O) applied to the Integrated Problem of Location Assignment and Automated Vehicles Scheduling (IPLAVS) in MCT at import. First, we present the problem and we demonstrate its NP-Hardness, then we introduce a modeling approach for the problem and finally we introduce a new cooperative Tabu search to solve the mono-objective problem and Multi-Objective Tabu Search Algorithm (MOTSA) adapted to solve efficiently combinatorial MOOP in general and Multi-objective IPLAVS in particular. We consider the three types of MACT.

Chapter 2

Vehicle Scheduling Problem in Automated Maritime Container Terminals

2.1. Introduction

Container terminals play a crucial role in global logistic networks. Because of the ever-increasing quantity of cargo, terminal operators need solutions for different decisional problems. In the maritime terminal, at boat arrival or departure, we observe five main problems: the allocation of berths, the allocation of query cranes, the allocation of storage space, the optimization of stacking cranes work load (or storage bay organization) and the scheduling and routing of vehicles. A good cooperation between the different installations in the terminal is important in order to minimize the container handling time.

2.2. AGV Scheduling Problem

2.2.1. Introduction

In an automated container terminal using Automated Guided Vehicles (AGVs) Query Cranes (QCs) and Automated Stacking Cranes (ASCs), numerical solutions have become essential to optimize the operators' decisions. Many recent researches have discussed the optimization of ACT equipment scheduling using different approaches. In this paper, we propose three mathematical models and an exact resolution method for the QC-AGV-ASC planning, the problem of tasks in Maritime Automated Container Terminal (MACT). Our first objective is to minimize the makespan (the time when the last task is achieved), and the second is to minimize the number of required

vehicles.

In an automated container terminal (ACT), the time of handling operations depends on the interactions between the different storage equipments. Different researches are established to improve the handling systems performance. We deal in the following with the problems which consider our two objectives (minimizing the makespan and minimizing the AGV fleet size). These two objectives are treated in the AGV scheduling problem. The problem of AGV scheduling was treated in the general context of AGVS and in the particular context of ACT. AGVS is a materials handling system that uses automated vehicles which are programmed to achieve tasks between different manufacturing and warehouse stations . It represents a very important innovation in international transport and logistics. ACT is one of the most famous examples of AGVS.

Studies of AGVS optimization have different objectives: maximizing the throughput, maximizing the vehicle utilization, minimizing the inventory level, minimizing the transportation costs, and maximizing the space utilization.

AGVS mathematical models have to respect some conditions to eliminate the traffic problems. Approaches used in AGVS optimization can be classified into two kinds: analytical approaches and simulation-based approaches. Analytical methods are mathematical techniques such as queuing theory, integer programming, heuristic algorithms, and Markov chains. A number of analytical approaches to AGVS optimization have been proposed in the literature.

In the next parts, we propose solutions for three terminal layouts and we use Meersman's results [1] to improve the mathematical modeling and the quality of our numerical solutions. We propose a model with two objectives: the optimization of operating time for the QC-AGV-ASC handling tasks and the minimization of the number of vehicles to be used. We use Meersman's mathematical results to perform our modeling and resolution and we propose new models for the scheduling problem using a partial containers' order and resolving large problem instances. Different layouts of MACT can be studied. Meersman presents two possible port architectures: a simple layout with

static AGV traffic and a complex layout with multiple variables paths. The traffic layouts considered in this part are presented in chapter 1.

2.2.2. Data construction

Data construction is based on terminal architecture and the handling speed of the equipment. We use next parameters to generate data: The quay length, the bay (named also yard) length and the distance between quay and storage zone.

The AGV and ASC transfer speed combined with the terminal dimensions give a clear idea about the data that we need for our modeling and simulations.

2.2.3 Theoretical result

Meersman [1] used a strategy of partial order to solve large instances of the scheduling problem: the tasks of each ASC are totally ordered. The author has supposed a sufficient quantity of AGVs which can ensure an optimal schedule and he has concluded an important theorem.

"Define the assignment order Π as the order in which the containers are assigned to the AGVs as they pass the common point. Moreover, define a suborder Π_s as a subset of Π, such that if i is ordered before j in Π_s, then i is ordered before j in Π, for all i, j Є Π_s.

Theorem: For each ASC s Є S, consider an optimal schedule. Let Π_s denote the order in which ASC s handles its containers. Then there exists an optimal assignment order Π, such that Π_s is a suborder of Π.".

In next parts of this section, we consider that the number of AGVs is sufficient to complete an optimal schedule.

2.2.4 Mathematical models

We consider a total order for each set of ASC handling tasks and another total order for each set of QC handling tasks (we use models with buffer space of QC equal to 1). In other terms, for any

QC loading task or ASC unloading task, the successor and the predecessor are initially known. In the next part, we consider that the matrices $ASC_{i,j}$ and $QC_{i,j}$ are constant.

We define the following data for all the models of this part.

2.2.4.1 Constants

All data presented in this part are initially known.

V: The set of vehicles (Automated Guided Vehicles in this section)

|V|: The vehicle fleet size.

ASC: The set of ASCs (Automated Stacking Crane)

QC: The set of QCs (Query Crane)

|QC|: The number of QCs used.

C: The set of containers. C is also equal to the set of handling tasks considering that every container is associated to one and only one handling task such that a global handling task is composed of loading-unloading tasks, transfer tasks and stacking task.

|C|: The numbers of containers to transfer and store. |C| is also equal to the number of handling tasks.

QC(i): Quay Crane initially assigned to unload container i from the ship.

ASC(i): Automated Stacking Crane initially assigned to unload container i from the associated AGV and load it in its storage location.

$QC_{i,j}$: If container j is unloaded directly after container i by the same QC, $QC_{i,j} = 1$ else $QC_{i,j} = 0$. We consider this data initially known.

$ASC_{i,j}$: If container j is unloaded from AGV and transferred to its storage location directly after container i by the same ASC, $ASC_{i,j} = 1$ else $ASC_{i,j} = 0$. We consider this data initially known. In

fact, we apply Meersman's theorem and we choose the order of tasks for every ASC (ASC order must respect QC order) without changing makespan optimality.

$T_{QC(i),BE(i)}$: The travel time between the QC unloading position (QC unloading point) and bay entry associated with storage location of container i.

$T_{BE(i),A}$: The travel time between the entry of storage bay of container i and the final position at quay entry (point A).

$T_{A,QC(i)}$: The travel time between point A (quay entry) and unloading position under QC associated with container i.

S_i: The ASC transfer time of task i depending on ASC speed and on distance between ASC transfer point (at the bay entry) and the exact storage location where container i will be stacked.

S_{QC}: The time that QC needs to unload container from the ship.

S_{ASC}: The time that ASC needs to pick up container from AGV.

S_s: Safety waiting time to be respected by AGVs near QC loading position.

t_0: Start time.

2.2.4.2 Variables

All the data introduced in this part are variables.

$AGV_{i,j}$: Decision variable, if container j is handled directly after container i by the same AGV

$AGV_{i,j}$ =1 else $AGV_{i,j}$ = 0.

$t_1(i)$: The start time of task i.

$t_2(i)$: The completion time of task i.

2.2.4.3 Formulation of the number of used vehicles

Consider |C| the number of tasks and |V| the number of Vehicles (AGVs), then:

$$|C| - \sum_{i \in C} \sum_{j \in C} AGV_{i,j} = |V|$$

Proof: $\sum_{i \in M} \sum_{j \in M} AGV_{i,j}$ is equal to the number of containers (or tasks) having direct predecessor considering AGV transfer task, then $|C| - \sum_{i \in C} \sum_{j \in C} AGV_{i,j}$ is equal to the number of containers or tasks not having a direct successor. A task with no direct successor is a first task for some AGV, then the number of those tasks is equal to the number of AGVs.

2.2.4.4 One-path and multiple fixed paths Mathematical Model

$$\min \max\{t_2(i) | i \in C\} \quad (1)$$

Subject to:

$$\sum_{j \in C} AGV_{i,j} \leq 1, \ \forall i \in C \quad (2)$$

$$\sum_{j \in C} AGV_{j,i} \leq 1, \ \forall i \in C \quad (3)$$

$$\sum_{i \in C} \sum_{j \in C} AGV_{i,j} = |C| - |V| \quad (4)$$

$$AGV_{i,i} = 0, \ \forall i \in C \quad (5)$$

$$t_1(i) \geq t_0, \ \forall i \in C \quad (6)$$

$$t_1(j) + G(1 - AGV_{i,j}) \geq t_2(i) + T_{A,QC(j)}, \ \forall i, j \in C \quad (7)$$

$$t_2(i) \geq t_1(i) + S_{QC} + T_{QC(i),BE(i)} + S_{ASC} + T_{BE(i),A}, \ \forall i \in C \quad (8)$$

$$t_2(j) \geq t_2(i) - T_{BE(i),A} + T_{BE(j),A} + s_i, \ \forall i, j \in C \setminus ASC_{i,j} = 1 \quad (9)$$

$$t_1(j) \geq t_1(i) + S_{QC} + S_s, \ \forall i, j \in C \setminus QC_{i,j} = 1 \quad (10)$$

Line 1: The objective is to minimize the completion time of the last task. That objective is generally named makespan and is equal to the operating time.

Constraints 2 and 3: Limit the number of direct successors and direct predecessors, every container has one or zero direct successor and one or zero direct predecessor.

Constraint 4: If we use k AGVs , k containers will have exactly zero successors and k containers will have exactly zero predecessors because every AGV will have a first task and a last task(final task for it).Then for n containers, only (n-k) tasks will be succeeded and only (n-k) tasks will be preceded.

Constraint 5: No container can precede or succeed itself.

Constraint 6: No task can start before t_0.

Constraint 7: Relation between two successive tasks of an AGV. If container j is handled directly after containers i with the same AGV, then $AGV_{i,j} = 1$ and we have:

$$t_1(j) \geq t_2(i) + T_{A,QC(j)}$$

else the relation will be:

$$t_1(j) + G \geq t_2(i) + T_{A,QC(j)}$$

and that is true because G is sufficiently large.

Constraint 8: $t_2(i) \geq t_1(i) + S_{QC} + T_{QC(i),BE(i)} + S_{ASC} + T_{BE(i),A}$: the final time of any task is equal or greater than the start time of the task plus the travel time between QC and ASC plus the QC loading time plus the ASC loading time.

Constraint 9: For every i, j in C if $ASC_{i,j} = 1$ then we have

$$t2(j) - T_{BE(j),A} - S_{ASC} \geq t2(i) - T_{BE(i),A} + S_i$$

$t_2(j) - T_{BE(j),A} - S_{ASC}$ is the date when AGV transferring container j is served by ASC.

$t_2(i) - T_{BE(i),A} + S_i$ is the date when ASC terminates the transfer of container i and reaches the ASC unloading position to wait next container transfer task.

In other terms, if the two vehicles transferring containers i and j are served successively by the same ASC ($ASC_{i,j} = 1$), then the vehicle transferring container j has to wait for ASC until it terminates the transfer of container i and returns to the unloading position at bay entry.

Constraint 10: the difference between 2 successive QC tasks is greater than or equal to the loading time under QC plus the safety time.

2.2.4.5 Multiple variables paths mathematical model

$T_{BE(i),QC(j)}$ is the travel time between the bay entry of location associated with container i (where container i is unloaded from AGV by ASC) and the QC loading position at the quay considering QC associated with container j (where container j is loaded on AGV by QC). If we consider the first model presented in part 2.2.4.4, we replace constraints (7), (8) and (9) respectively by constraints (11), (12) and (13), presented next, to obtain the following modelling which is adapted to the layout of multiple variables paths.

$\min \max\{t_2(i) | i \in C\}$

Subject to:

Constraints (2), (3), (4), (5), (6) and (10) of the static traffic model

$$t_1(j) + G(1 - AGV_{i,j}) \geq t_2(i) + T_{BE(i),QC(j)}, \ \forall i, j \in C \quad (11)$$

$$t_2(i) \geq t_1(i) + S_{QC} + T_{QC(i),BE(i)} + S_{ASC}, \forall i \in C \quad (12)$$

$$t_2(j) \geq t_2(i) + S_i + S_{ASC}, \forall i, j \in C \setminus ASC_{i,j} = 1 \quad (13)$$

2.2.4.6 Bi-objective model

To solve correctly the scheduling problem using the theorem of sub-orders, we need to use a sufficient number of AGVs for the optimal schedule. This number will depend on the routing path distances, the AGV transfer speed, ASC transfer speed and QC unloading speed. For our optimization approach, we consider that the makespan has higher priority than AGV fleet size, then in our modelling we can naturally use the theorem of sub-orders proved by Meersman because the minimal numbers of AGVs that we search has to satisfy the operating time optimality.

In 2001, IFA's team developed a minimum flow algorithm to determine the number of AGVs required at a semi-automated container terminal [6]. Our bi-objective model is a good alternative to solve the scheduling problem in a short run time giving small numbers of required AGVs. Considering multiple variables paths model, we replace (1) by (14) and (4) by (15), and then we obtain a new model which is more efficient and more intelligent. This model has two objectives: minimize the completion time of the last task and minimize the number of AGVs necessary to complete the optimal scheduling. Constraints (2), (3), (5), (6), (10), (11), (12) and (13) of multiple variables paths layout are used in this model.

$$\min \left(G \max\{t_2(i) | i \in C\} + \left(|C| - \sum_{i \in C} \sum_{j \in C} AGV_{i,j}\right) \right) \quad (14)$$

Subject to:

Constraints (2), (3), (5), (6), (10), (11), (12) and (13)

$$\sum_{i \in C} \sum_{j \in C} AGV_{i,j} \leq |C| - 1 \quad (15)$$

G is a sufficiently large number which insures that makespan has higher priority than the number of AGVs in the optimization process.

2.3 Auto-Strad scheduling problem

2.3.1 Introduction

In this part, we propose three mathematical models and an exact resolution of QC-Auto-Strad planning, the problem of tasks in MACT with Automated Straddle-Carriers (Auto-Strad).

We use Meersman's results [1], as in part 2.2, to improve the mathematical modeling and resolution efficiency. We suggest modeling with two objectives: the optimization of operating time and the minimization of Auto-Strad fleet size. We consider models of terminal architecture presented in figures 11 and 12 using the same partial containers' order and solving equivalent problem instances as in part 2.2.

2.3.2 Data construction

Data generation is based on terminal architecture and handling speed of equipment. The main difference between the data considered in part 2.2 and data considered in part 2.3 concerns essentially routing paths and vehicle speed.

2.3.3 Theoretical result

Theoretical result (Meersman's theorem) of part 2.2.3, is considered in this part. Naturally, security waiting time, at bay entry, is not dependent on storage locations.

In the following part of current section, we assume that the number of Auto-Strads is sufficient to complete an optimal schedule.

2.3.4 Mathematical Models

We consider a total order for each set of storage tasks in common bay and another total order for each set of QC tasks (we use models with buffer space of QC equal to 1). We consider that the matrices $B_{i,j}$ and $QC_{i,j}$ are constant. We define the following data for all the models.

2.3.4.1 Constants

All the data introduced in the following are initially known.

C, V, QC, $QC_{i,j}$, QC(i), $T_{QC(i),BE(i)}$,$T_{BE(i),A}$, $T_{A,QC(i)}$, $S_{QC,}$ S_s and t_0 are defined as in section 2.2

B: The set of storage bays

$B_{i,j}$: If container j is stored directly after container j in the same bay $B_{i,j}$= 1, else $B_{i,j}$ =0. This data is initially known. In fact, we apply Meersman's theorem and we choose the order of transfer tasks for every bay (the order of entering the storage bay must respect QC unloading order).

BE(i): The entry of storage bay where is the location associated with container i.

S_i: The Auto-Strad routing time between bay entry and storage location assigned to container i.

S_V: The time that Auto-Strad needs to load or unload container.

S_2: Safety waiting time near bay entry.

2.3.4.2 Variables

All data presented in this part are variable.

$V_{i,j}$: Decision variable, if container j is handled directly after container i by the same Auto-Strad $V_{i,j}$ =1, else $V_{i,j}$ = 0

t_1(i): The start time of handling task i

t_2(i): The completion time of handling task i such that task i is composed of loading task under QC, routing task and stacking task at storage location.

2.3.4.3 Formulation of the number of used vehicles

We consider the same formulation of part 2.2.4.3.

2.3.4.4 One-path and multiple fixed paths mathematical Model

We present next the mathematical model of the Auto-Strad scheduling problem considering the modelling of part 2.2.4.4.

$$\text{Min } \max\{t_2(i) | i \in C\} \quad (01)$$

Constraints (2) to (7) and constraint (10) of section 2.2.4.4

$$t_2(i) \geq t_1(i) + 2\,S_V + T_{QC(i),BE(i)} + T_{BE(i),A}, \forall i \in C \quad (16)$$

$$t_2(j) \geq t_2(i) - T_{BE(i),A} + S_V + S_2 + T_{BE(j),A},\ \forall i, j \in C \setminus B_{i,j} = 1 \quad (17)$$

Constraint 16: the final time of any task is equal or greater than its start time plus the travel time between QC and storage location plus the Auto-Strad loading time under QC plus the Auto-Strad stacking time at storage location.

Contraints 17: $\forall i, j \in C |\ B_{i,j} = 1$: $t_2(j) - T_{BE(j),A} \geq t_2(i) - T_{BE(i),A} + S_V + S_2$

$t_2(j) - T_{BE(j),A}$ is the date when Auto-Strad transferring container j enters the storage bay. $t_2(i) - T_{BE(i),A}$ is the date when Auto-Strad transferring container i enters the storage bay. If the two vehicles enters successively the same storage bay ($B_{j,i} = 1$), then vehicle transferring container i has to respect a waiting time of $S_V + S_2$ seconds at bay entry. In fact when an Auto-Strad accesses a storage bay, that one is blocked during $S_V + S_2$ seconds and the next vehicle cannot access the bay before the end of that period.

For the one-path and multiple variables paths mathematical model we do exactly the same modifications as in section 2.2.3.5 with S_B instead of S_j and S_v instead of $S_{ASC.}$ Considering the bi-objective modelling of section 2.2.4.6, we do exactly the same changes to obtain the bi-objective model for MACT using Auto-Strads.

2.4 ALV scheduling problem

We consider all results of part 2.2 with nil waiting time at ASC unloading position ($S_i = 0$). For new modelling, constraint 9 in part 2.3 is removed and the ASC loading time S_{ASC} in constraint 8 is replaced by the ALV unloading time S_{ALV}.

2.5 Numerical results

We choose CPLEX optimizer to test the performance of our models. The application of the sub-orders theorem combined with the use of constraint (4) give the possibility to solve instances of hundreds of containers but with a use of a number of vehicles more than 10 percent of the containers number. Using the third model, we can solve the scheduling problem with a small number of vehicles because the model has two objectives: minimize containers handling and transfer time and minimize the number of vehicles to be used. We solve problem instances of 10 to 500 containers with a GAP of 0.15 to 0 percent. One of our most important results is the resolution of the bi-objective problems (minimizing handling time and vehicle resources) of 500 containers, 3 QCs and 8 ASCs (or storage bays for straddle carrier case). The GAP is not stable, the vehicle and crane speeds and the paths routing time for some instances can increase the GAP value. With the first presented model, using sufficient number of AVs (between 10 and 15 percent of the tasks numbers) we resolve small and big problem instances with optimal solution. The third model (two-objective model) is more efficient for the instance with a limited number of vehicles. Results depend on the layout model: for the static traffic layout problem instances with less than 150 containers are generally easily solved and the two objectives are reached with double optimality.

2.5.1 Results for MACT with AGVs

Results of bi-objective modeling in the static traffic case

Instance*	Makespan GAP	Fleet size GAP	Total GAP	Run time
150/3/6	0%	0%	0%	4 s
250/4/12	0%	0%	0%	6 s
300/4/12	0%	0%	0%	6 s
500/3/8	0%	0%	0%	25 s
500/4/8	0%	> 0%	0.11%	60 s

(*) Instance: number of containers / number of QCs / number of ASCs

Table 3

Results of bi-objective modeling with the multiple variables paths layout

Instance*	Makespan gap	Fleet size gap	Total gap	Run time
150/3/6	0%	0%	0%	10 s
150/4/12	0%	0%	0%	10 s
200/4/12	0%	0%	0%	12 s
300/4/8	0%	0%	0%	15 s
300/4/12	0%	> 0%	0.15%	70 s

Table 4

Comparison of presented bi-objective model to Meersman's model

	Presented Modelling	Meersman's model
Objective(s)	Two objectives: minimizing makespan minimizing AGV fleet size	One objective: minimizing makespan
Equipment	QC-AGV-ASC	QC-AGV-ASC
Performance	A gap of 0 % for instances up to 500 containers, 4 QCs and 12 ASCs. For these instances the runtime is between 0 seconds and 60 seconds.	A gap of 0 % to 8 % for instances up to 170 containers 27 ASCs and 24 AGVs. For these instances the runtime is between 0 seconds and 658 seconds.
Conditions	Consider a sufficient resource of AGVs. Consider the QC handling task as a unique constant independent from container location in the ship.	Consider the QC handling task as a constant dependent on container.

Table 5

2.5.2 Results for MACT with ALVs

Results of bi-objective modeling in the static traffic case

Instance*	Makespan GAP	Fleet size GAP	Total GAP	Run time
150/3/6	0%	0%	0%	3 s
250/4/12	0%	0%	0%	5 s
300/4/12	0%	0%	0%	10 s
500/3/8	0%	0%	0%	40 s
500/4/8	0%	> 0%	0.02%	80 s

Table 6

(*) Instance: number of containers / number of QCs / number of ASCs

Results of bi-objective modeling with the multiple variables paths layout

Instance*	Makespan gap	Fleet size gap	Total gap	Run time
150/3/6	0%	0%	0%	4 s
150/4/12	0%	0%	0%	20 s
200/4/12	0%	0%	0%	18 s
300/4/8	0%	0%	0%	32 s
300/4/12	0%	> 0%	0.05%	30 s

Table 7

2.5.3 Results for MACT with Auto-Strads

Results of bi-objective modeling with single path layout

Instance*	Makespan GAP	Fleet size GAP	Total GAP	Run time
150/3/6	0%	0%	0%	2 s
250/4/12	0%	0%	0%	9 s
300/4/12	0%	0%	0%	10 s
500/3/8	0%	0%	0%	44 s
500/4/8	0%	> 0%	0.01%	59 s

Table 8

(*) Instance: number of containers / number of QCs / number of storage bays

Results of bi-objective modeling with multiple paths layouts

Instance*	Makespan gap	Fleet size gap	Total gap	Run time
150/3/6	0%	0%	0%	6 s
150/4/12	0%	0%	0%	22 s
200/4/12	0%	0%	0%	31 s
300/4/8	0%	0%	0%	48 s
300/4/12	0%	> 0%	0.03%	30 s

Table 9

2.6 Conclusion

A new generation of terminal using automated container handling equipment needs solutions to optimize task scheduling and operating costs. Many storage strategies, statistical studies, mathematical models and algorithms are proposed by researchers. To solve the planning of QC-AV-ASC, we present an effective model for every kind of traffic layout. We propose an efficient bi-objective model, which is important to determine the optimal storage time and the minimal number of AVs (Automated Vehicles) required. The bi-objective model can solve large instances (until 500 containers) with double optimality (giving the optimal makespan and the minimum number of required AVs) in reasonable run time (less than 60 s). To the most of our knowledge, our bi-objective model is the first model optimizing in one time the makespan and the AV fleet size in automated container terminal. Our models consider three handling equipments (AV, QC and ASC) which is an efficient approach. We treat the three existing AVs at MACT: AGVs, ALVs and Auto-Strads.

Chapter 3

Integrated Problem of Location Assignment and Vehicle Scheduling in Automated Maritime Container Terminals at Import

In this part we propose a new integrated modeling by considering the import case in Maritime Automated Container Terminals (MACT). We consider combination between two known problems, the first is the storage location assignment problem and the second is the straddle carrier scheduling problem. In fact, we study the Multi-Objective Integrated Problem of Location Assignment and Vehicle Scheduling (IPLAVS) in MACT at import. This approach which combines two chronologically successive problems leads to the use of multi-objective optimization (MOO).

The objective is to minimize the operating cost which we evaluate by considering eight components: the date of last task noted "makespan", the total vehicle operating time, the total storage bay occupation time, the number of vehicles used, the number of storage bays used, the number of storage locations used, and two different costs of storage location assignment. The location assignment costs are evaluated in order to facilitate the containers transfer for deliveries. We assume that the operating cost is a function of these components and that the influence of each component is variable and dependent on different parameters. These parameters are essentially: the number of quays in the terminal, the straddle carrier traffic layout, the number of container ships to serve in the terminal, the influence of concurrent operations in the terminal, the storage space configuration, the number of free storage bays, the number of free straddle carriers, the number of free quay cranes (QCs), the mobility of quay cranes; etc.

In this part, we study an integrated problem, which combines the equipment allocation, and scheduling and the location assignment in MACT at import. These two problems are generally treated separately. This combination improves the productivity of the handling system due to better theoretical optimality. The only study of the integrated problem of storage space allocation and vehicle scheduling, in the general context of container terminals, was the study of Bish et al. [20] which treats the problem for automated container terminals (AGV handling system). In this work, the vehicle schedule and location assignment are optimized in order to minimize one objective, which is the handling time. However, the waiting-times in bay entry (AGV wait for stacking crane in bay entry), which is a crucial constraint of the real problem, is not considered. In other studies, the optimization of storage location assignment in container terminal considers total vehicle routing distance. However, vehicle scheduling, waiting time in bay entries, as well as the interaction between the different equipment and others parameters are not considered.

In our study, we consider the multi-objective aspect of the problem with eight realistic objectives to optimize, in which it is a new and efficient approach considering the state of art. We treat with the following objectives: the makespan (date of last task or operating time), the number of straddle carriers used, the sum of straddle carrier operating times, the sum of storage bay occupation times, the number of storage bays used, the number of storage locations used and two location costs.

3.1 Integrated Problem of Location Assignment and Straddle Carrier Scheduling in Automated Maritime Container Terminals at Import

3.1.1 Operating process

In this section we consider the Integrated Problem of Location Assignment and Straddle Carrier Scheduling in Maritime Container Terminal at import (IPLASS). The layout of a container terminal influences seriously the straddle carrier productivity. In fact, the number and the dimensions of storage bays, the number and the length of quays and the number of quay cranes affects the quality of vehicle traffic, specifically when considering the routing time.

The traffic layout is another parameter, which influences the straddle carrier operating time. This parameter has a second impact, which concerns the complexity of the problem. Figure 14 presents the terminal layout considered in our study.

In this work, we present a new model for IPLASS. We consider general terminal layout regarding the following properties:

- ✓ Only one quay is considered. Then, for multiple quays terminal, we consider the problem only for container ships allocated to the same quay.
- ✓ Considering QC (Quay Crane) scheduling, the container-unloading schedule is initially known for each QC. In fact, we have to determine only straddle carrier schedule and storage location assignment. Considering our layout choices, when storage location assignment is determined, the straddle carrier assignment and schedule are naturally identified (see the end of this part).
- ✓ Our modeling supports multiple container-ship unloading operations considering compatible arrival times. To support multiple-ship instances, it is crucial to know the exact date of container ship arrivals, the number of the QCs used and the schedule of container handling for each QC.
- ✓ The vehicle routing in the quay must respect a unique direction.
- ✓ Straddle carriers picks up containers under QCs respecting a known global schedule. This schedule is based on the sub-schedules of containers unloading from ships (QC schedule). If n is the number of QCs used, for every n successive transfer tasks, straddle carriers have to serve all QCs; in other words, QCs are served successively by the fleet of vehicles (note that vehicles are straddle carriers for all this chapter). QCs operate loading tasks in parallel and do not wait for vehicles arriving. Theoretically, this scheduling constraint influences the problem feasibility and optimality.
- ✓ We consider that the vehicles are initially in the preloading position near QCs.
- ✓ Every handling task begins when a straddle carrier picks up the container under QC.

- ✓ Every handling task ends when a straddle carrier stacks the container in the associated location.
- ✓ Considering the straddle carrier traffic, we define the quay entry (Point A in figure 14) and the quay exit (Point B in figure 14) as the entry and the output of the part of the quay reserved for QCs used.
- ✓ After picking up containers under quay cranes, the straddle carrier moves to quay output, then it moves to the entry of the bay where the storage location associated to the container is. It waits enough to avoid an accident with the last vehicle entering the same bay, and finally it transfers the container to its storage location in the bay. After the end of this handling task, the vehicle moves to the quay entry and then it moves to the position of the next container picking up task under QC.
- ✓ At anytime of the process, it is easy to determine which straddle carrier to use for the next container-handling task. In fact, we only have to choose the first vehicle returning to the quay entry (Point A in figure 14).
- ✓ Considering the storage space, the set of free storage locations is initially known, however, we have to determine the storage locations to use for stacking containers. The free storage bays are naturally determined by the free storage locations, and the storage bays used are determined by the storage locations used for each solution to the problem.
- ✓ With the chosen layout, each handling task is naturally identified by the associated container.

For experiments we use real databases of "Terminal de Normandie" in the Maritime Port of Le Havre. The terminal is presented in Figure 15.

Figure 15 - CT "Terminal de Normandie" « source: Google Maps »

3.1.2 Mathematical modeling

In the following, we present the mathematical model of the multi-objective IPLASS in maritime terminal at import. It integrates new and realistic constraints which reflect the real functioning of the terminal. We develop constraint formulations to insure an efficient location for the set of containers in order to facilitate some tasks, like the next transportation, the deliveries or the storage of next arriving containers. These constraints are associated to the evaluation of two location assignment costs. The constraints that are associated with the first location assignment cost are essential to facilitate the container transfer to the next transporter (or the delivery) minimizing the total distance among containers of same customer or of same delivery date, while those associated to the second one are used to maximize for each storage location its possibilities to receive containers taking into account their delivery dates and the delivery date of the last container stored in that location.

Our objective is to solve the problem considering the real need of the decision maker which is the minimization of the real global operating cost. This quantity is mostly considered as a linear function of different components. In some situations, weighted sum methods are efficient to solve MOOP. Concerning the case of multi-objective combinatorial problems, using linear objective function, we cannot propose some efficient solutions to the user because the Pareto Front Region (PFR) is non-convex.

Many parameters can influence the operating cost, but the eight chosen objectives represent the most important cost components in the IPLASS in a maritime terminal at import. Operating cost in

the general context of MCT concerns essentially storage space resources, equipment resources and operating time resources. Consider now the handling system in container terminal managed by straddle carriers. Storage space resources are bays and locations. Equipment resources are the quay cranes and the straddles carriers. Operating time resources concerns firstly the makespan, which is the date of the last container-handling task (picking up, transfer and storage); secondly the sum of storage bay occupation times and thirdly the sum of straddle carrier-operating times. Considering these resources, there is concurrence between the different operations in the terminal and especially for the multi-quay terminals and when operators have to serve many container ships or other transport vehicles at one time. This concurrence influences the weight of the different operating cost components. A global approach can be a good response to this problem of operating cost evaluation. In fact, we can solve the problem for many container ships at one time considering a global weighted cost. For the number of free straddle carriers, we can consider that the terminal uses a sufficient number of vehicles to satisfy every container ship. However, the waiting times under QCs make the total number of straddle carriers used in the terminal limited by an upper bound which we evaluate in part 3.1.5. Considering this limit, operator decision has to take into account the concurrence between container ships for QC resource. If the terminal has a limited number of Straddle Carriers, the concurrence between container ships for vehicles is significant for MOO.

The initial storage space configuration is another parameter which influences the operating cost. It is also an important factor determining the density of feasible solutions in the solution space and the lower bound of the number of storage bay used. The lower bound of storage bays resource is a very important factor influencing the makespan lower bounds and the resolution hardness. In fact, with a small number of storage bays used, the total straddle carriers waiting time at the bay entry increases considerably. The number of free storage bays in the terminal at the container ship(s) arrival influences directly the cost of storage space resources and the adequate objective weight.

In our approach, we consider that the operating cost is a vector of eight objective evaluations.

3.1.2.1 Data

QC: The set of Quay Cranes used.

QC(i): The Quay Crane associated to container i. For each container i, QC(i) is initially known.

C: The set of tasks (or containers). We can identify each task by its container by considering the known total order of container picking up process.

B: The set of free storage bays available for use for stacking containers.

B(p): The bay of storage location p.

BE(p): Bay entry of storage location p.

P: The set of storage locations. Every location has an initial capacity.

w(p): The initial storage capacity of location p. It is the number of free levels of p. The storage capacity at every location depends on the initial configuration of the storage space. Consider a terminal with a storage space of k levels, if the storage location p contains n containers, then for the next handling operation at container ship arrival or departure, the capacity of p is equal to k – n.

S_{QC}: QC unloading time. It is the time that QC needs to unload container from the ship. S_{QC} is considered static.

S_v: Container storage time. When Auto-Strad arrives to the storage location, S_v is the static time which the straddle carrier needs to stack the container in the associated storage location. It is also considered as the time which Auto-Strad needs to load container under QC.

S_B: Maximal security waiting time in bay entry. S_B is static security parameters. Every straddle carrier has to wait for at most S_B seconds in the bay entry. The condition ($S_B > S_v$) is crucial to eliminate accidents between vehicles entering the same storage bay with an arrival difference less than S_v.

$succ_{QC}(i)$: The direct successor of container i considering picking up task under QC.

$T_{p,QC(i)}$: Straddle carrier routing time from storage location p to QC associated with container i.

$T_{QC(i),BE(p)}$: Straddle carrier routing time from QC associated with container i to bay entry of storage location p.

T_p: Transfer time between the entry of the bay, where is the storage location p, and the exact position of p.

d_i: Delivery date of container i to its next transporter (or customer).

d_p: Delivery date of the last container stored in location p.

G: a sufficiently big number.

C(i): the set of containers having the same next transporter or the same delivery date (considering the day of delivery) as container i.

T(x,y): Routing time between the entry of storage bay x and the entry of storage bay y. This parameter is used to evaluate the first location assignment cost $f_{i,j,x,y}$.

$f_{i,j,x,y}$: The containers allocation cost associated with the decision which stacks container i in storage bay x and container j in storage bay y. $f_{i,j,x,y}$ is initially known data. Note that $f_{i,j,x,y}$ is defined for i Є C and j Є C(i).

$f_{i,j,x,y} = T(x,y)$ if $(x \neq y)$

$f_{i,j,x,y} = (|QC|-1)\ S_B$ if $(x=y)$

3.1.2.2 Variables

V: The set V represents vehicles used (vehicles are straddle carriers in this section). It is an order used to specify every vehicle. |V| is the straddle carrier fleet's size. We consider that |V| can be as large as necessary. |V| is an objective to minimize in the optimization problem.

$V = \{1, 2, \ldots, |V|\}$.

B^*: The set of storage bays used for stacking containers. We have to use exactly all these bays. B^* is determined by the storage locations decision. $|B^*|$ is the size of B^*, and it is an objective to minimize

in the optimization problem.

P^*: the set of storage location used for storing container after decision.

v_i: Straddle carrier assigned to container i. $v_i \in V$.

$X_{i,p}$ is equal to1 if container i is stacked in storage location p, else $X_{i,p}$ is equal to 0.

$X'_{i,p}$ is equal to 1 if container i is the first container stored in location p considering the current handling operation, otherwise $X'_{I,p}$ is equal to 0.

$V_{i,j}$ is equal to1 if container j is transferred directly after the container i by the same vehicle, otherwise $V_{i,j}$ is equal to 0. This variable is defined for $i \in C$ and $i \neq j$.

$P_{i,j}$: 1 if container j is stored directly after i in the same location (in others terms, container j is stored on container i).

$P'_{i,j}$: Binary variable, equal to1 if and only if container j is stored in the same location as container i, directly or indirectly after i (container j is stored on container i, but other containers can be stored between i and j). $P'_{i,j}$ is defined for $i \neq j$.

$B_{i,j}$: 1 if containers i and j are stacked in the same bay and container j is stacked directly after i considering the stacking order in the bay. $B_{i,j}$ is defined only for $i \neq j$.

$t_1(i)$: Start time of task i. The date when the associated straddle carrier picks up container i under QC.

$t_2(i)$: The date when straddle carrier assigned to container i accesses the storage bay of chosen location.

$t_3(i)$: Completion time of task i. The date when associated straddle carrier stores container i in its storage location.

t_v: Termination time of vehicle v (straddle carrier v) considering all containers associated to v.

$t_{b:}$ Termination time of container storage in bay b. We consider all containers assigned to storage bay b.

C_{Max}: The makespan which is the date of the last handling task.

I_p: If the storage location p is used I_p is equal to 1, otherwise I_p is nil.

I_b: If the storage bay b is used I_b is equal to 1, otherwise I_b is nil.

$h_{i,j}$: Equal to $d_j - d_i$ if container j is stored on container i, 0 else.

$g_{i,p}$: Equal to $d_p - d_i$ if container i is the first container stored in p after making decision, 0 else.

$F^7_{i,j}$: a function which evaluates partially the first location assignment cost, equal to zero if containers i and j have note the same client, else equal to the routing time between locations assigned to i and j.

$Z_{i,j}$: Equal to 1 if the location decision assigned to containers i and j will cause an unproductive move, 0 else. These variables are used to evaluate the number of unproductive moves to be caused by location assignment decision, considering all containers except those to be stored in first free levels of each location.

$Z'_{i,j}$: Equal to 1 if i is the first container stored in location p and causes an unproductive move considering the initial storage space configuration. These variables are used to evaluate the number of unproductive moves to be caused by location assignment decision, considering only containers to be stored in the first free levels of each location.

3.1.2.3 Modeling

Objective

Minimize

$$\begin{pmatrix} \mathbf{C_{Max}} \\ |\mathbf{C}| - \sum_{\mathbf{i}\in\mathbf{C}} \sum_{\mathbf{j}\in\frac{\mathbf{C}}{\mathbf{i}}} \mathbf{V_{i,j}} \\ \sum_{\mathbf{b}\in\mathbf{B}} \mathbf{I_b} \\ \sum_{\mathbf{p}\in\mathbf{P}} \mathbf{I_p} \\ \sum_{\mathbf{v}\in\mathbf{V}} \mathbf{t_v} \\ \sum_{\mathbf{b}\in\mathbf{B}} \mathbf{t_b} \\ \frac{\sum_{\mathbf{i}\in\mathbf{C}} \sum_{\mathbf{j}\in\mathbf{C},\mathbf{j}\neq\mathbf{i}} \mathbf{F_{i,j}} / |\mathbf{C(i)}|}{|\mathbf{C}|} \\ \sum_{\mathbf{i}\in\mathbf{C}} \sum_{\mathbf{j}\in\mathbf{C},\mathbf{j}\neq\mathbf{i}} \mathbf{Z_{i,j}} + \sum_{\mathbf{i}\in\mathbf{C}} \sum_{\mathbf{p}\in\mathbf{P}} \mathbf{Z_{i,p}} \end{pmatrix}$$

Location constraints

$$\forall \mathbf{i} \in \mathbf{C}: \ \sum_{\mathbf{p}\in\mathbf{P}} \mathbf{X_{i,p}} = \mathbf{1} \quad \textbf{(01)}$$

$$\forall \mathbf{p} \in \mathbf{P}: \sum_{\mathbf{i}\in\mathbf{C}} \mathbf{X_{i,p}} \leq \mathbf{w(p)} \quad \textbf{(02)}$$

Vehicle scheduling constraints

$$\forall \mathbf{i} \in \mathbf{C} \ \sum_{\mathbf{j}\in\mathbf{C/i}} \mathbf{V_{i,j}} \leq \mathbf{1} \quad \textbf{(03)}$$

$$\forall \mathbf{i} \in \mathbf{C}: \sum_{\mathbf{j}\in\mathbf{C/i}} \mathbf{V_{j,i}} \leq \mathbf{1} \quad \textbf{(04)}$$

$$|\mathbf{C}| - \sum_{\mathbf{i}\in\mathbf{C}} \sum_{\mathbf{j}\in\mathbf{C/i}} \mathbf{V_{i,j}} \geq \mathbf{1} \quad \textbf{(05)}$$

Constraints about schedule of container transfer in storage bay

$$\forall \mathbf{i} \in \mathbf{C}: \sum_{\mathbf{j}\in\mathbf{C}\setminus\mathbf{i}} \mathbf{B_{i,j}} \leq \mathbf{1} \quad \textbf{(06)}$$

$$\forall \mathbf{i} \in \mathbf{C}: \sum_{\mathbf{j}\in\mathbf{C/i}} \mathbf{B_{j,i}} \leq \mathbf{1} \quad \textbf{(07)}$$

$$\forall (\mathbf{p}, \mathbf{i}) \in \mathbf{P} \times \mathbf{C}: \ \mathbf{I_p} \geq \mathbf{X_{i,p}} \quad \textbf{(08)}$$

$$\forall \mathbf{b} \in \mathbf{B}, \mathbf{I_b} \leq \sum_{\mathbf{p}\in\mathbf{b}} \mathbf{I_P} \quad \textbf{(09)}$$

$$\forall \mathbf{p} \in \mathbf{P}: \mathbf{I_{B(p)}} \geq \mathbf{I_p} \quad \textbf{(10)}$$

$$|\mathbf{C}| - \sum_{(\mathbf{i},\mathbf{j})\in\mathbf{C}\setminus\mathbf{i}\neq\mathbf{j}} \mathbf{B_{i,j}} = \sum_{\mathbf{b}\in\mathbf{B}} \mathbf{I_b} \quad \textbf{(11)}$$

$$\forall \mathbf{b} \in \mathbf{B}, \forall (\mathbf{i}, \mathbf{j}) \in \mathbf{C^2}: \ \mathbf{B_{i,j}} + \mathbf{B_{j,i}} \leq \frac{\sum_{\mathbf{p}\in\mathbf{b}} \mathbf{X_{i,p}} + \sum_{\mathbf{f}\in\mathbf{b}} \mathbf{X_{j,f}}}{\mathbf{2}} \quad \textbf{(12)}$$

Transfer time constraints

$$\forall (\mathbf{i}, \mathbf{j}) \in \mathbf{C^2}, \mathbf{j} = \mathbf{succ_{QC}(i)}: \mathbf{t_1(j)} \geq \mathbf{t_1(i)} + \mathbf{S_{QC}} \quad \textbf{(13)}$$

$$\forall (\mathbf{i}, \mathbf{p}) \in \mathbf{C} \times \mathbf{P}: \ \mathbf{t_2(i)} \geq \mathbf{t_1(i)} + \mathbf{S_V} + \mathbf{T_{QC(i),BE(p)}} + \mathbf{G}\left(\mathbf{X_{i,p}} - \mathbf{1}\right) \quad \textbf{(14)}$$

$$\forall (\mathbf{i}, \mathbf{j}) \in \mathbf{C^2}, \mathbf{i} \neq \mathbf{j}: \ \mathbf{t_2(j)} \geq \mathbf{t_2(i)} + \mathbf{S_B} + \mathbf{G(B_{i,j}} - \mathbf{1}) \quad \textbf{(15)}$$

$$\forall (\mathbf{i}, \mathbf{e}) \in \mathbf{C} \times \mathbf{P}: \mathbf{t_3(i)} \geq \mathbf{t_2(i)} + \mathbf{T_e} + \mathbf{S_v} + \mathbf{G(X_{i,e} - 1)} \quad \textbf{(16)}$$

$$\forall (\mathbf{i}, \mathbf{j}) \in \mathbf{C^2}, \forall \mathbf{p} \in \mathbf{P}: \mathbf{t_1(j)} \geq \mathbf{t_3(i)} + \mathbf{T_{p,QC(j)}} + \mathbf{G(V_{i,j} + X_{i,p} - 2)} \quad \textbf{(17)}$$

$$\forall \mathbf{i} \in \mathbf{C}: \mathbf{C_{Max}} \geq \mathbf{t_3(i)} \quad \textbf{(18)}$$

Constraints about vehicle attributions and termination time

$$\forall \mathbf{i} \in \mathbf{C}, \mathbf{1} \leq \mathbf{v_i} \leq |\mathbf{C}| - \sum_{\mathbf{i} \in \mathbf{C}} \sum_{\mathbf{j} \in \mathbf{C/i}} \mathbf{V_{i,j}} \quad \textbf{(19)}$$

$$\forall (\mathbf{i}, \mathbf{j}) \in \mathbf{C^2}, \quad \mathbf{v_i - v_j \leq G(1 - V_{i,j})} \quad \textbf{(20)}$$

$$\forall (\mathbf{i}, \mathbf{j}) \in \mathbf{C^2}, \quad |\mathbf{v_i - v_j}| \geq \mathbf{1 - V_{i,j}} \quad \textbf{(21)}$$

$$\forall (\mathbf{v}, \mathbf{i}) \in \mathbf{V} \times \mathbf{C}, \mathbf{t_v} \geq \mathbf{t_3(i) - G|v - v_i|} \quad \textbf{(22)}$$

Constraint about storage bay termination time

$$\forall \mathbf{b} \in \mathbf{B}, \ \mathbf{t_b} \geq \mathbf{t_3(i)} - \mathbf{G}(\mathbf{1} - \sum_{\mathbf{p} \in \mathbf{b}} \mathbf{X_{i,p}}) \quad \textbf{(23)}$$

Constraint about location costs

$$\forall \mathbf{i} \in \mathbf{C}, \mathbf{j} \in \mathbf{G(i)}, \forall (\mathbf{b_1}, \mathbf{b_2}) \in \mathbf{B^2}: \mathbf{F_{i,j}^7} \geq \mathbf{f_{i,j,b_1,b_2}} - \mathbf{G}(\mathbf{2} - \sum_{\mathbf{p} \in \mathbf{b_1}} \mathbf{X_{i,p}} - \sum_{\mathbf{l} \in \mathbf{b_2}} \mathbf{X_{j,l}}) \quad \textbf{(24)}$$

$$\forall (\mathbf{i}, \mathbf{j}) \in \mathbf{C^2}, \mathbf{i} \neq \mathbf{j}, \mathbf{p} \in \mathbf{P}, \ \mathbf{P_{i,j} + P_{j,i} \leq (X_{i,p} + X_{j,p})/2} \quad \textbf{(25)}$$

$$\forall \mathbf{i} \in \mathbf{C}, \ \sum_{\mathbf{j} \in \mathbf{C} \setminus \mathbf{i}} \mathbf{P_{i,j}} \leq \mathbf{1} \quad \textbf{(26)}$$

$$\forall \mathbf{i} \in \mathbf{C}, \ \sum_{\mathbf{j} \in \mathbf{C} \setminus \mathbf{i}} \mathbf{P_{j,i}} \leq \mathbf{1} \quad \textbf{(27)}$$

$$|\mathbf{C}| - \sum_{(\mathbf{i},\mathbf{j}) \in \mathbf{C} \setminus \mathbf{i} \neq \mathbf{j}} \mathbf{P_{i,j}} = \sum_{\mathbf{p} \in \mathbf{P}} \mathbf{I_p} \quad \textbf{(28)}$$

$$\forall (\mathbf{i}, \mathbf{j}) \in \mathbf{C^2} \setminus \mathbf{i} \neq \mathbf{j}: \ \mathbf{t_3(j)} > \mathbf{t_3(i)} + \mathbf{G(P_{i,j} - 1)} \quad \textbf{(29)}$$

$$\forall (\mathbf{i}, \mathbf{j}) \in \mathbf{C^2}, \mathbf{i} \neq \mathbf{j}, \mathbf{d_j} > \mathbf{d_i}: \ \mathbf{Z_{i,j} \geq G(P'_{i,j} - 1)} \quad \textbf{(30)}$$

$$\forall (\mathbf{i}, \mathbf{p}) \in \mathbf{C} \times \mathbf{P}, \mathbf{d_i} > \mathbf{d_p}: \mathbf{Z'_{i,p} \geq 1 + G(X_{i,p} - 1)} \quad \textbf{(31)}$$

$$\forall (\mathbf{i}, \mathbf{j}) \in \mathbf{C^2}, \mathbf{i} \neq \mathbf{j}: \ \mathbf{P'_{i,j}} = \mathbf{P_{i,j}} + \sum_{\mathbf{k} \in \mathbf{C}} \left\lfloor \frac{\mathbf{P_{i,k} + P_{k,j}}}{\mathbf{2}} \right\rfloor + \sum_{\mathbf{f} \in \mathbf{C}} \sum_{\mathbf{h} \in \mathbf{C}} \left\lfloor \frac{\mathbf{P_{i,f} + P_{f,h} + P_{h,j}}}{\mathbf{3}} \right\rfloor \quad \textbf{(32)}$$

⌊ X ⌋ : Integer part of real X.

3.1.2.4 Objectives

a) Straddle Carriers' Makespan

The straddle carriers' makespan is the date of completion of the last vehicles task. We denote this objective by C_{Max}. The straddle carriers' makespan is a crucial parameter to qualify the Pareto solutions. The straddle carrier makespan is the global makespan of the handling system. The last straddle carriers' task is operated when the last container is stored in the storage space. Considering realistic data, high quality or optimality of QC-makespan is insured by straddle carrier makespan optimality.

Consider solution S, T_{last} its last routing time and S_{last} its last waiting time at bay entry. Consider T_{min} and T_{max}, the minimal and the maximal routing time for each container transfer task respectively.

$$\text{Makespan}(S) = \text{QC Makespan}(S) + T_{last} + S_v + S_{last}$$

S_{last} the Straddle Carrier's waiting time in bay entry considering the last storage task. S_{last} is equal at least to 0 and at most to (|QC|-1) S_B.

$$T_{min} + S_v \leq T_{last} + S_v + S_{last} \leq T_{max} + S_v + (|QC| - 1) \times S_B$$

$$\text{QC Makespan } (S) \geq \text{Makespan}(S) - (T_{max} + S_v + (|QC| - 1) \times S_B)$$

$$\text{QC Makespan } (S) \leq \text{Makespan}(S) - (T_{min} + S_v)$$

The variation of QC Makespan(S) is equal to $T_{max} + S_{max} - (T_{min} + S_{min})$.

When Makespan(S) is optimal, QCs' Makespan(S) has good quality. In fact, with Straddle Carrier makespan optimality, QC makespan is equal at most to:

$$\text{Makespan}(S) - (T_{min} + S_v).$$

We can optimize QCs' makespan considering its natural minimization with straddle carriers' makespan minimization.We can also attribute a particular location for the last container stored to insure optimality of QCs' makespan when straddle carriers' makespan is optimal. We denote by L_{last}

this location. The routing path associated to L_{last} must be minimal and its storage bay cannot be used for other container storage. With these conditions, we insure the equation "$\text{QC Makespan}(S) = \text{Makespan}(S) - (T_{min} + S_v)$"and the optimality of straddle carriers' makespan and QCs' makespan are equivalent.

b) The number of vehicles used

The number of vehicles to be used is denoted by |V|. When we optimize IPLASS, we consider that the terminal has a large resource of straddle carriers, and can use for each handling operation at container ship arrival a sufficiently large number of vehicles to insure makespan high quality or optimality. The only one upper bound, which concerns the number of straddle carriers used, is the natural parameters of QC productivity (showed in part 3.1.5). We evaluate |V| as below:

$|\boldsymbol{V}| = |\boldsymbol{C}| - \sum_{\boldsymbol{i}\in\boldsymbol{C}} \sum_{\boldsymbol{j}\in\boldsymbol{C/i}} \boldsymbol{V_{i,j}}$ (part 3.1.3.4).

If makespan optimality has highest priority, the optimal straddle carrier fleet size is the smallest number which satisfies next condition:

For an optimal solution of the problem, when QC finishes unloading a container from a ship, at least one vehicle is ready to pick it up under the QC.

Determining this optimal straddle carrier fleet size is studied by IRIS F.A. VIS [18].

c) The number of storage bays used

The number of storage bays is an important parameter to qualify the operator decision in SCMT. In fact, at multi-ship arrival, the work in the terminal is organized in different handling operations, and the goal of everyone is to transfer containers from a specific part of the quay to the storage space. Considering our modeling, every handling operation concerns one or many container ships but a unique part of the quay. Every handling operation is specified by the associated part of the quay and its entry and exit points (Point A and Point B in Figure 14).

A concurrence between the handling operations concerns especially the storage locations' assignment. When the operator assigns a set of storage locations to containers, a set of storage bays is used. Handling operations cannot use storage bays at the same time without communication. If

the frequency of storage bay occupation by straddle carriers is high, the operator cannot use the same storage bays for different handling operations at the same time. Then, it is important to minimize the number of storage bays used for every handling operation. We denote this variable by |B*| and we evaluate it as below:

- ✓ $|\boldsymbol{B}^*| = \sum_{\boldsymbol{b}\in\boldsymbol{B}} \boldsymbol{I_b}$

The number of storage bays used must respect next equation:

- ✓ $|\boldsymbol{B}^*| = |\boldsymbol{C}| - \sum_{\boldsymbol{i}\in\boldsymbol{C}} \sum_{\boldsymbol{j}\in\boldsymbol{C/i}} \boldsymbol{B_{i,j}}$

d) The number of storage locations used

When different handling operations can use the same storage bays, may be they cannot use the same storage locations in each common bay. The use of a common storage location by different handling operations during a common operating time depends on communication quality and storage strategy. However, the minimization of the number of locations used is an efficient parameter to qualify the operator's decision. We denote this quantity by |P*| and we evaluate it as below:

- ✓ $|\boldsymbol{P}^*| = \sum_{\boldsymbol{p}\in\boldsymbol{P}} \boldsymbol{I_p}$

The number of storage locations used has to respect the next equation.

- ✓ $|\boldsymbol{C}| - \sum_{\boldsymbol{i}\in\boldsymbol{C}} \sum_{\boldsymbol{j}\in\boldsymbol{C/i}} \boldsymbol{P_{i,j}} = |\boldsymbol{P}^*|$

e) The total operating time of straddle carriers - $\sum_{\boldsymbol{v}\in\boldsymbol{V}} \boldsymbol{t_v}$

When we minimize the makespan, the operating time is globally optimized. However, if we consider the operating time of every straddle carrier, we have to add another objective which is the sum of vehicles' operating time. That objective is evaluated as below:

- ✓ $\sum_{\boldsymbol{v}\in\boldsymbol{V}} \boldsymbol{t_v}$

f) The total occupation time of storage bays - $\sum_{\boldsymbol{b}\in\boldsymbol{B}} \boldsymbol{t_b}$

When a storage bay is used for a handling operation, the decider has to consider its occupation

time. After this time the storage bay can be easily used for another handling operation. Then we consider the objective of minimizing the sum of storage bay occupation time by the current operation. That objective is noted and evaluated as below:

- ✓ $\sum_{b \in B} t_b$

g) The first Location Assignment Cost (LAC_1)

Consider the known data C(i) ,which is the set of containers having the same delivery date and the same next transporter as container i. We evaluate $F_7(i)$ as an average cost considering every container j in C(i), $f_{i,j,x,y}$ and |C(i)| the size of C(i).

- ✓ $\forall i \in C\ ,\ F_7(i) = \sum_{j \in C \setminus i} F_{i,j}^7 / |C(i)|$.

Considering F7(i) for every container i and the number of containers |C|, we evaluate the first Location Assignment Cost $F_7(C)$.

- ✓ $F_7(C) = (\sum_{i \in C} F_7(i))/|C|$.

For a real instance of the problem, average evaluation of location cost gives a better idea about the quality of the global location decision. The optimization of the first Location Assignment Cost is essential to promote the facilitation of container transfer to the next transporter.

h) The second Location Assignment Cost (LAC_2)

The second Location Assignment Cost is equal to the number of unproductive moves caused by location decision. These unproductive moves will disadvantage the facility of container deliveries. An unproductive move is caused when a container with some delivery date is stored on another container with earlier delivery date.

3.1.2.5) Constraints

(01) Each container i is to be stacked in exactly one storage location.

(02) The number of containers to stack in each storage location p is less than or equal to the initial capacity of this location considering the beginning of the current handling operation.

(03) Every container has at most one direct successor considering straddle carrier handling task.

(04) Every container has at most one direct predecessor considering straddle carrier handling task.

(05) The number of straddle carriers used is at least equal to one.

(06) Every container has at most one direct successor considering transfer and stacking tasks in the same bay.

(07) Every container has at most one direct predecessor considering transfer and stacking tasks in the same bay.

(08) For each storage location p if at least for one container the location decision variable $X_{i,p}$ is equal to 1 then I_p is equal to 1 and p is used. Then, the storage bay p is used if and only if at least one location decision variable $X_{i,p}$ is equal to 1. In other terms, the storage location p is used if and only if at least one container is stored in p.

(08) If no storage location contained by the storage bay b is used, then b is not used.

(09) If storage location p is used, then the storage bay B(p) which contains p is used.

(09)+(10) Storage bay b is used if and only if at least one storage location in b is used.

(11) The number of used storage bays is equal to the sum of I_b. This constraint fixes the number of variable $B_{i,j}$ equal to 0, which is the number of used bays.

(12) Antecedence and succession for transfer and stacking tasks in storage bays concern only containers stacked in storage locations of the same bay.

(13) Suppose that container j is the direct successor of container i considering straddle carrier picking up task under QC, then the start time of handling operations of j (denoted by $t_1(j)$) is at least equal to start time of handling tasks of i (denoted by $t_1(i)$) added to the picking up time of container under QC denoted by S_{QC}.

(14) The date when straddle carrier associated to container i enters the associated storage bay (the storage bay access date) denoted by $t_2(i)$ is at least equal to the start date of handling tasks of i noted by t1(i) plus the time that Auto-Strad needs to load container plus the period of routing from the QC

which unloaded i to the entry of associated storage bay noted by $T_{QC(i),BE(p)}$.

(15) Consider a container i and its direct successor for storage task in the same bay. Then the storage bay access date of j denoted by $t_2(j)$ is at least equal to the storage bay access date of i $t_2(i)$ added to security time S_B. This constraint insures the condition of possible waiting time when straddle carrier enters the storage bay. It is a security constraint used to eliminate accidents in bays.

(16) The termination time of handling tasks of container i denoted by $t_3(i)$ is equal to the date when straddle carrier accesses the storage bay entry added to T_e, the routing time between the bay entry and the storage location e associated to i, plus the container loading time Sv.

(17) If container j is the direct successor of container i considering straddle carrier picking up task, then the start date of handling tasks of j (t1(j)) is at least equal to the completion time of handling tasks of i (t3(i)) added to the straddle carrier routing time between the storage location p associated to i (Tp,QC(j)) and the QC associated to j (QC(j)).

(18) The makespan is superior to completion time of every handling task.

(19) Determination of the index set which represents the set of straddle carriers used. If V is the set of vehicles used, then for each container i: $v_i \in \{ 1 , 2 ,\ldots, |V| \}$ (v_i is the vehicle assigned to container i).

(20) If the straddle carrier precedence variable $V_{i,j}$ is equal to one then the containers i and j are transferred by the same vehicle ($v_i=v_j$).

(19) If the straddle carrier precedence variable $V_{i,j}$ is equal to zero then the straddle carrier associated to i is not the same as the one associated to j.

(20)and (21) For every container i and j, vehicle vi is equal to vehicle v_j if and only if $V_{i,j}$ is equal to 1.

(22) Termination date of vehicle v (straddle carrier v), denoted by t_v, is higher than termination date of each container storage task, considering the containers transferred by v.

(23)Termination date of storage task in each bay b is upper than termination date of storage of each container stacked in b.

(24) Evaluation of location cost $f_{i,j,x,y}$ considering location decisions and their associated storage bays.

(25) For each container i and j stored in different locations, the variable $P_{i,j}$ is nil.

(26) Each container i has at least one direct successor considering stacking task in the same storage location.

(27) Each container i has at least one direct predecessor considering stacking task in the same storage location.

(28) The number of containers which have no predecessor (or the number of containers which have no successor) considering the stacking tasks in the same storage location is equal to the number of storage locations used. In fact, the first (or the last) container stored in each storage location has no predecessor (or successor) considering the current handling operation.

(29) If the decision variable $P_{i,j}$ is equal to one, the container i is stored before the container j in the same location ($t_3(j) > t_3(i)$).

(30) $Z_{i,j}$ is binary number equal to one if and only if container j is stored on container i and will be delivered after container i.

(31) $Z'_{i,p}$ is binary number equal to one if and only if container i is stored in location p before any other container (we take into account containers to unload for current ship arrivals), and will be delivered after the last container stored in p during precedent storage operations (considering precedent ship arrivals).

(32) $P'_{i,j}$ is binary number equal to one if and only if container j is stored on container i directly or indirectly. That variable is used to evaluate the number of unproductive moves to be caused by location decision. Note that at most four containers can be stored in the same storage location at MACT using Straddle-Carriers.

3.1.3 NP-Completeness of IPLASS

In this part, we prove the NP-Hardness of IPLASS. Consider the instance I of IPLASS:

$$|C| = |P|, \forall (i, j) \in C^2 : d_i = d_j, \forall (p, f) \in P^2 : B_p \neq B_f, S_{QC} = 0 \text{ and } S_B = 0$$

For this instance, containers have the same delivery date, then they are identical considering the final storage space configuration which determines the second Location Assignment Cost for next operations. The sum of free storage location is equal to the number of containers, and then the number of bays to use is initially known. The number of free straddle carriers is less than the optimal number of straddle carriers, then all straddle carriers are necessary to insure makespan optimality. Considering this instance, IPLASS can be modeled as:

$$\begin{aligned} &Minimize\ C_{Max} \\ &\forall\ v \in V, \sum_{p \in P} X_{v,p}\ t_p \leq C_{Max} \\ &\forall\ p \in P, \sum_{v \in V} X_{v,p} = 1 \end{aligned}$$

$X_{v,p}$ is equal to 1 if the vehicle v has to transfer a container to the storage location p.

t_p is the routing time of a vehicle which has to transfer a container to the storage location p. t_p depends only on p.

Consider now the parallel machines problem (PMP) modeled as next:

$$\begin{aligned} &Minimize\ C_{Max} \\ &\forall m \in M, \sum_{s \in T} Y_{m,s}\ \gamma_s \leq C_{Max} \\ &\forall s \in T, \sum_{m \in M} Y_{m,s} = 1 \end{aligned}$$

M: set of parallel machines. T: set of tasks. γ_s : processing time of task s.

$Y_{m,s:}$ equal to 1 if task s is assigned to machine m and equal to 0 else.

Each feasible schedule of instance I can be translated to a feasible schedule of the PMP with the same makespan. In fact, the set of vehicles of instance I corresponds to the set of parallel machines

because there is no interaction between straddle carriers. In others terms, the total routing time of every vehicle does not depend on any other straddle carrier.

Every vehicle of instance I is associated to a machine of the PMP. For each transfer task, the routing time of the vehicle depends only on the storage location associated to the container. Routing time of vehicle v is the processing time of the machine associated to v in parallel machine problem. Every feasible schedule of I can be translated to a feasible schedule of PMP with the same makespan.

Consider now the parallel machine problem. We construct a particular instance I of the IPLASS.

$|C| = |P|, \forall (i, j) \in C^2 : d_i = d_j, \forall (p, f) \in P^2 : B_p \neq B_f, S_{QC} = 0 \text{ and } S_B = 0$

Every machine in parallel machine problem is associated to a vehicle in I. Every task s in parallel machine problem is associated to a storage location p in I. The processing time of task s is equal to vehicle routing time necessary to transfer a container to the associated location p.

Reciprocally, every feasible schedule of PMP can be translated to a feasible schedule of i with the same makespan. We conclude that the instances I are NP-Completes. Then we know that the problem is NP-Complete considering handling operation using many straddle carriers. We will prove now that IPLASS is NP-Complete even for instances using only one vehicle.

Consider next instance of IPLASS (instance J):

$$|V| = 1, |P| = |C|, \forall (x, y) \in P: B_x \neq B_y$$

The problem is now modeled as follows:

$$Min \sum_{\substack{i \in C \\ x \in P}} \sum_{\substack{j \in G(i) \\ y \in P}} (f_{i,j,x,y} \times X_{i,x} \times X_{j,y})$$

$$\forall i \in C, \sum_{p \in P} X_{i,p} = 1$$

$$\forall p \in P, \sum_{i \in C} X_{i,p} = 1$$

For this instance of IPLASS, we have only to solve the containers allocation problem minimizing the first Location Assignment Cost. In fact, we obtain a quadratic assignment problem. The new problem has some particularities compared to the known quadratic allocation problems. To prove NP-Hardness of instances J, we use polynomial reduction of J to a known NP-Complete problem which is the Traveling Salesman Problem (TSP).

Firstly we add to instance J next properties: the vehicle speed is equal to 1 unity.

Let's consider the set of containers $C = \{C_i: 1 \leq i \leq n\}$, $n=|C|$. Containers are listed as following: $(C_1, C_2, \ldots, C_n)$. For each $i \leq n$, we have the next relationship: C_i and C_{i+1} have the same delivery date or the same customer, but we cannot have the two relationships at the same time. In fact, if C_i and C_{i+1} have the same delivery date, they have not the same customer. On the other hand, if they have the same customer they have not the same delivery date. Then, for every container C_i we have: $C(i) = \{C_{i-1}, C_{i+1}\}$. The new problem is to assign, for every container C_i, a storage location $P(C_i)$ minimizing the sum of quantities $T(P(C_i,), P(C_{i+1}))$,

with $i \leq n-1$.($T(x,y)$ is the routing time between the locations x and y).

$$\sum_{1 \leq i \leq n-1} T\big(P(C_i), P(C_{i-1})\big)$$

For the chosen instance, vehicle speed is equal to 1 and all used storage bays are different. Then $T(P(C_i),P(C_{i+1}))$ is equal to the distance between $P(C_i)$ and $P(C_{i+1})$ for every container $i \leq n-1$. Minimizing the sum of distances between every container location and the location of its successor considering the chosen assignment leads to solving the Traveling Salesman Problem (TSP). In fact every storage location in instance J can be considered as a city in TSP. The decision $P(C_i)$ in instance J is equivalent to choose the next city to visit after i-1 decisions in TSP.

Reciprocally, each instance of TSP is equivalent to an instance J_2 (instance having the properties of J). Actually, we can associate the set of cities in TSP to a set of storage location in J_2 considering a fictive container terminal where the distance between every two free storage locations is equal (or proportional) to the distance between the two corresponding cities. The traveler in TSP is associated to the vehicle in J and we have $|V|=1$. In TSP, the cities are different and the storage locations in J_2

are in different bays:

$$\forall (x, y) \in P: B_x \neq B_y$$

In TSP, the traveler visits each city exactly one time, then the associated vehicle in J loads (visit) every storage location exactly one time. In other words, instance J_2 has the two next properties:

$$|P| = |C|$$

Then instance J_2 has all the properties of J and each instance of TSP is equivalent to an instance in J.

We conclude that instances J of IPLASS are NP-Complete and then IPLASS is NP-Complete even considering a straddle carrier resource limit of 1 vehicle. The problem is NP-Complete considering large part of its instances.

NP-Completeness of IPLASS is a direct result of NP-Completeness of two IPLASS sub-problems which are the location assignment problem and the straddle carrier scheduling problem.

3.1.4 Evaluation of solution quality

3.1.4.1 Evaluation of |V| Upper Bound and |V| Lower Bound

|V| is the straddle carrier fleet size. We define the routing cycle as the vehicle routing path including routes from point A to QC, from QC to storage location and from storage location to point A. The maximal number of vehicles used depends on two factors. The first factor is the maximal productivity of loading tasks under QCs which depends on their speed and their number. The second is the maximal operating cycle, which is the sum of the pick-up time under QC, the maximal routing cycle, and the maximal waiting time in the bay entry. We note T_{Max} the maximal operating cycle and S_{Max} the maximal waiting time in bay entry. S_{Max} depends on the number of QCs and on S_B. S_B is the maximal waiting time between each straddle carrier and its direct predecessor vehicle operating in the same bay. When a straddle carrier v arrives in bay entry, the bay can be blocked by one or more vehicles which preceded v in that bay. When v enters the storage bay, the bay is blocked for S_B fixed security time. We consider now that $S_{QC,}$ the pick-up time under QCs, is bigger than S_B. We denotes by $S(v_1,v_2)$ the waiting time in bay entry between a

vehicle v_1 and its direct predecessor v_2. We notes $t_B(v_1)$ and $t_B(v_2)$ the respective arrival times of v_1 and v_2 in bay entry.

$S(v_1,v_2) = \max (0, S_B - (t_B(v_1)- t_B(v_2))$

Then, there is no waiting time in bay entry between vehicles coming from the same QC. In fact, if v_1 and v_2 are served by the same QC, $t_B(v_1)- t_B(v_2) > S_{QC}>S_B$ than $S(v_1,v_2)=0$. The result is that the waiting time in bay entry exists only between vehicles coming from different QCs with equivalent routing starting times from QCs. Consider $t_{QC}(v_1)$ and $t_{QC}(v_2)$ these routing starting times from QCs.

$S(v_1,v_2) = \max (0, S_B - (t_{QC}(v_1)- t_{QC}(v_2))$

Consider now a vehicle v and all straddle carriers which proceeded v in the same storage bay. If a vehicle v_1 coming from QC_1 affects the total waiting time of v in bay entry no other vehicle coming from QC_1 can affect it because of pick-up delay under QC_1 which results from the last equation. Then, at most |QC|-1 predecessor can affect the total waiting time of v in bay entry. We note it S(v) and we note $\{ v_i / i< |QC| \}$ the set of straddle carriers which preceded v in the same storage bay and come from different QCs.

$$S(v) = \sum_{1 \le i < |QC|} S(v, v_i)$$

$$S(v) \le (|QC| - 1) \times S_B$$

We conclude that maximal waiting in bay entry S_{Max} is equal to $(|QC| - 1) \times S_B$.

$$T_{Max} = S_{QC} + Max\{T_{A,p} + T_{p,A} : p \in P \} + S_{Max}$$

We have finally an evaluation of the vehicle fleet size upper bound.

$$|V| \le \frac{T_{Max}}{S_{QC}} \times |QC|$$

We note that with sufficiently large stacking capacity under QCs the QCs' productivity is maximal. In other terms, this condition insures the fact that when the QC unloads container from ship, it does not wait for the straddle carriers to unload containers from the stacking space under it.

Suppose now two conditions:

The stacking space under QCs is sufficiently large and QCs productivity is maximal.
The C_{Max} Optimality is an absolute priority.

Considering these two conditions, we can evaluate a lower bound for the number of straddle carriers used considering solutions with optimal makespan. Consider S_{opt} a solution having an optimal makespan, the minimal number of vehicles used for S_{opt} depends on three factors: the QCs handling speed, which depends on S_{QC}, the number of QCs and the minimal operating cycle T_{Min}. We evaluate T_{Min} as:

$$T_{Min} = Min\{T_{A,p} + T_{p,A} : p \in P\} + S_{QC}$$

Considering these conditions, we have next the inequality:

$$|V(S_{opt})| \geq \frac{T_{Min}}{S_{QC}} \times |QC|$$

Consider now the last two conditions added to the next proprieties.

We suppose the existence of C_{Max}-Optimal solution S_{opt} with nil waiting times in bay entries.
There is no possibility of short cut paths between the quay entry (point A) or the quay exit (point B) and the storage bay. All storage bays have the same length. In other terms, all straddle carrier routing paths are equal.

With these conditions, the inequality becomes an equality and we obtain a strict evaluation of $|V(S_{opt})|$-Optimality.

$$\left|V(S_{opt})\right| = \frac{T_{Min}}{S_{QC}} \times |QC|$$

3.1.4.2 Evaluation of C_{Max} Lower-Bound

For this part we note Makespan(S) the value of C_{Max} considering solution S. It is the date of the last task for the decision S. Consider V* the minimal set of vehicles sufficient to insure makespan optimality.

In our modeling, for each vehicle, the routing path includes the routing path from the QC

unloading the container, to the storage location and the routing path from the storage location to the QC associated with the next container to transfer. In this part, we consider that routing path include the routing path from point A (quay entry) to the storage location and the routing path from the storage location to point A. This consideration does not affect the nature of the problem and its optimality. Consider p_0 the storage location with the shortest routing cycle. We note T_0 the routing time of the shortest routing cycle added to loading time at QC and unloading and stacking time at storage location. We note N_0 the maximal number of containers which the largest fleet of straddle carriers can transfer in the routing time T_{Min}. In mathematical terms we can note:

$$T_{Min} = Min\{T_{A,p} + T_{p,A} : p \in P\} + S_{QC} + S_B$$

$$N_0 = |QC| \frac{T_0}{S_{QC}}$$

With |QC| the cardinal of QCs set, A the point at quay entry and $T_{A,p}$ the routing time between A and storage location p and $T_{p,A}$ the routing time between p and A.

We consider the storage location p associated to the minimal total routing path:

$$Min\{T_{A,k} + T_{k,A} : k \in P\}.$$

We consider w(v,i) the vehicle waiting time in bay entry. It is exactly the waiting time for vehicle v when it transfer container i.

V^* is the minimal set of vehicles which insure maximal productivity for the speed of containers picking up under QCs. V^* is evaluated considering that waiting times in bay entry are nil. Then we can conclude that $|V^*|=N_0$.

Consider a solution S, we have:

$$Makespan(S) \geq \frac{|C|}{|V^*|} \times T_{Min}$$

Proof

$$
\begin{aligned}
Makespan\ (S) &= Max\left\{ \sum_{i\in C(v),\ j=succ_v(i)} \left(T_{QC(i),\,p(i)} + T_{p(i),QC(j)} + S_{QC} + S_V + w(v,i)\right); v \in V^* \right\} \\
&\geq Max\left\{ \sum_{i\in C(v)} \left(T_{A,\,p(i)} + T_{p(i),A} + S_{QC} + S_v \right); v \in V^* \right\} \\
&\geq Max\left\{ \sum_{i\in C(v)} Min\left\{ T_{A,\,p(i)} + T_{p(i),A} + S_{QC} + S_v \ ; i \in C(v) \right\}; v \in V^* \right\} \\
&\geq Max\left\{ \left|C(v)\right| Min\left\{ T_{A,p} + T_{A,p} + S_{QC} + S_v \ ;\ p \in P \right\}; v \in V^* \right\} \\
&\geq Max\left\{ \left|C(v)\right| ; v \in V^* \right\} \times Min\left\{ T_{A,p} + T_{p,A} + S_{QC} + S_v \ ;\ p \in P \right\} \\
&\geq \frac{|C|}{|V^*|}\, T_{Min}
\end{aligned}
$$

Consider now that the stacking space under QCs has sufficiently large capacity to insure QCs' operating time optimality. With that condition we can evaluate the date of the last QC handling task with the next formulation. We denote this quantity Makespan(QC_{opt}) and we note C(q) the set of containers associated to QC q.

$$Makespan\left(QC_{opt}\right) = Max\{\ |C(q)| \times\ S_{QC}\ ,\qquad q \in QC\}$$

Considering the same conditions we evaluate another lower bound for the makespan of the global QC-SC handling system. Suppose that all routing paths correspond to the same routing time T. S is a feasible solution to the problem.

$$Makespan(S) \geq Makespan\left(QC_{opt}\right) + T$$

In our modeling, we consider a regular QC unloading task with a static container unloading time less than the straddle carrier picking up time SQC. A total container picking up schedule is initially

considered. For the most general context, we evaluate the next inequality for each solution S and for each QC handling situation.

$$Makespan(S) \geq \frac{|\boldsymbol{C}|}{|\boldsymbol{QC}|} \times S_{QC} + (|C| mod |QC|) \times S_{QC}$$

Consider now the set of storage bays used B^*. We note b the storage bay in B^* containing the largest set of storage locations used. We suppose the productivity of straddle carriers maximal. We note T, the routing time associated with each container transfer to b. In these conditions we have the following equation:

$$Makespan(S) = (|b| - 1) \times S_B + T$$

B^* and b are initially unknown. At least b contains $|C|/|B^*| + |C| \bmod |B^*|$ containers. Suppose that all routing paths correspond to the same routing time T. Then, for each solution S, we have next inequality.

$$Makespan(S) \geq \left(\frac{|C|}{|B^*|} - 1\right) \times S_B + |C| mod |B^*| + T$$

3.1.4.3 Evaluation of |B*| Lower Bound

The lower bound, considering the number of bays used, is equal to the cardinal of the smallest set of bays which contains a set of free locations with a total storage capacity equal to the number of containers to stack. In mathematical terms we note the next evaluation of $|B^*|$-Lower Bound.

$$L.B(|B^*|) = Min\{\, |F|; F \subset B \text{ and } \sum_{b \in F} \sum_{p \in b} w(p) = |C| \,\}$$

3.1.4.4 Evaluation of |P*| Lower Bound

The lower bound, considering the number of used storage locations, is equal to the cardinal of the smallest set of locations having a total storage capacity equal to the number of containers to stack. In mathematical terms we note next the evaluation of $|P^*|$-Lower Bound.

$$L.B(|P^*|) = Min\{\, |H|; H \subset P \text{ and } \sum_{p \in H} w(p) = |C| \,\}$$

3.1.4.5 Evaluation of $\sum t_v$ Lower Bound

Consider P^*_{Min} the set of storage locations to be used. We suppose that P^*_{Min} contains the |C| storage locations which corresponds to the |C| shortest routing paths. For each solution S we have the following inequality:

$$\sum\nolimits_{v\in V} t_v \geq \sum_{p\in P^*_{Min}} (T_{A,p} + T_{p,A} + S_{QC} + S_v)$$

Suppose that every routing path have the same length T, then we obtain the next result.

$$\sum\nolimits_{v\in V} t_v \geq |C| \times (T + S_{QC} + S_v)$$

3.1.4.6 Evaluation of $\sum t_b$ Lower Bound

We note Tb the routing time between the entry of the bay b and the quay entry (Point A). Consider the storage bay b in B*, we know that $t_b \geq (|b| - 1) \times S_B + T_b$ (part 3.1.5) then we can conclude next inequality.

$$\sum_{b\in B^*} t_b \geq (|C| - |B^*|) \times S_B + \sum_{b\in B^*_{Min}} T_b$$

Suppose that all routing paths have the same length which corresponds to the operating time T. We obtain the next result.

$$\sum_{b\in B^*} t_b \geq (|C| - |B^*|) \times S_{\mathrm{B}} + |B^*| \times T$$

3.1.4.7 Evaluation of Lower Bound for the first Location Assignment Cost

The minimal distance between two different storage locations is a lower bound of the first Location Assignment Cost.

3.1.4.8 Evaluation of Lower Bound of the second Location Assignment Cost

Considering the second Location Assignment Cost, the lower bound is equal to zero.

3.2 Integrated Problem of Location Assignment and ALV Scheduling in Maritime Container Terminal at import

3.2.1 Operating process

Compared to the case of Auto-Strad traffic, operating process in MACT using AGVs has next particularities:

- ✓ AGV cannot access the storage bays, it is the ASC which unloads the container from the vehicle (the AGV) and transfers it to its exact storage location.
- ✓ At bay entry, the waiting times of vehicles are dependent on ASC handling operations; in fact the ASC cannot serve any AGV until finishing the transfer of last served container to its storage location, storing it and back to the unloading position at bay entry.
- ✓ When the ASC unloads the container from AGV, the vehicle (the AGV) returns the quay entry to begin the next transfer task.

3.2.2 Mathematical modeling

3.2.2.1 Data

All data defined in this part are initially knowns.

We define QC, QC (i), C, P, w (p), $S_{QC,}$ d_i, G, $T_{QC(i),BE(p)}$ and C (i) as in part 3.1.2.1.

ASC: The set of free Automated Stacking Cranes available for use to transfer and stack containers.

S_{ASC}: Container storage time. When an ASC arrives to the storage location, S_{ASC} is the static time which the ASC needs to stack the container in the associated storage location.

S_p: The time which the ASC needs to transfer a container from the transfer point (ASC unloading point at bay entry) to the storage location p, store the container in the associated location and

return from the location p to the transfer point.

$succ_{QC}(i)$: The direct successor of container i considering picking up task under QC.

$T_{BE(p),QC(i)}$: ALV routing time from ASC unloading position at bay entry associated with storage location p to QC associated with container i.

3.2.2.2 Variables

All data defined in this part are variables.

V, $V_{i,j}$,v_i, $X_{i,p}$, $X'_{i,p}$, $P_{i,j}$, $P'_{i,j}$, $Z_{i,j}$, $Z'_{i,j}$, I_p, t_v and C_{max} are defined as in part 3.1.2.2.

ASC^*: The set of ASCs used for stacking containers. We have to use exactly all these ASCs. ASC^* is included in ASC. ASC^* is determined by the location decision. $|ASC^*|$ is the size of ASC^* and is an objective to minimize in Multi-Objective IPLAVS.

$ASC_{i,j}$: 1 if containers i and j are stacked by the same ASC and container j is stacked directly after i. $ASC_{i,j}$ is defined only for $i \neq j$.

$t_1(i)$: Start time of task i. The date when the ALV picks up container i under QC (considering our chosen layout, we can also consider the date when the ALV move from point A without changing the optimality of the problem).

$t_2(i)$: The date when ALV assigned to container i releases i at the bay entry and exactly at the unloading position of associated ASC .

t_h : Termination time of ASC h. We consider all containers assigned to storage bay b.

I_h: If ASC h is used I_h is equal to 1, otherwise I_h is nil.

L(h): The set of storage locations associated with ASC h.

We consider the same model as in part 3.1 with some modifications:

- ✓ Instead of B, the set of storage locations we consider ASC, the set of Automated Stacking Cranes.
- ✓ Instead of S_B, the static blocking period in the case of Auto-Strad traffic, we consider Se

which is a waiting time dependent on the storage location e assigned to the considered container.

- ✓ Instead of Te,QC(i) the routing time between storage location e and QC associated with container i, we consider

3.2.2.3 Modelling

In this part, we present mathematical modeling of multi-objective IPLAVS in automated maritime terminal with ALVs at import. It integrates new and realistic constraints which reflect the real functioning of the terminal.

For mathematical modelling of IPLAVS in the case of MACT using ALVs as vehicles, we consider constraints (01) to (15) and (19) to (32) of section 3.1.2.3 with minor modifications:

- ✓ For constraints (06) and (07), we consider $ASC_{i,j}$ instead of Bi,j, where $ASC_{i,j}$ is binary variable equal to one if and only if container j is transferred directly after container i with the same ASC (Automated Stacking Crane).
- ✓ In constraints (10) and (11), instead of B the set of storage bay we consider ASC the set of Automated Stacking Cranes and instead of I_b we use I_h which is a decision variable equal to one if and only if the ASC h is used. We use also the relation $p \in L(h)$ instead of $p \in b$, where L(h) is a given data equal to the set of storage locations associated with the ASC h.
- ✓ For constraints (12) and (13), we consider ASC instead of B, $ASC_{i,j}$ instead of $B_{i,j}$ and $p \in L(h)$ instead of $p \in b$.
- ✓ For constraint (19), the makespan denoted C_{max} is evaluated with $t_2(i)$ instead of $t_3(i)$.
- ✓ In constraints (24), instead of t_b and $t_3(i)$ we consider, respectively, th, the date of operating termination of ASC h and $t_{ASC} (i)$, the date when ASC terminates transfer task of container i and return to loading position at bays' entry. In addition, we consider the relation $p \in L(h)$ instead of $p \in b$.
- ✓ In constraint (25), we consider $(h_1, h_2) \in ASC^2$ instead of $(b_1, b_2) \in B^2$, $p \in L(h_1)$ instead

of $p \in b_1$ and $l \in L(h_2)$ instead of $p \in b_2$.

- ✓ For constraint (30), we consider $t_{ASC}(i)$ and $t_{ASC}(j)$ instead of $t_3(i)$ and $t_3(j)$.We add also next constrains instead of constrains (16), (17) and (18):

$$\forall (i,j) \in C^2, i \neq j: \; t_{ASC}(j) \geq t_{ASC}(i) + S_p + G(ASC_{i,j} + X_{j,p} - 2) \quad \textbf{(33)}$$

$$\forall (i,e) \in C \times P: t_{ASC}(i) \geq t_2(i) + S_p + G(X_{i,p} - 1) \quad \textbf{(34)}$$

$$\forall (i,j) \in C^2, \forall p \in P: t_1(j) \geq t_2(i) + T_{ASC(p),QC(j)} + G(V_{i,j} + X_{i,p} - 2) \quad \textbf{(35)}$$

(33) Consider a container i and its direct successor j considering transfer task by the same ASC. The date of ASC transfer termination for j is at least equal to the date of ASC transfer termination of i added to the ASC transfer and return times considering storage location assigned to container j.

(34) The termination time of ASC handling tasks of container i (denoted by $t_{ASC}(i)$) is at least equal to the date when ALV accesses ASC loading point added to the routing time between that point and the storage location assigned to i and the ASC return time from the storage location to the unloading position (ASC transfer point).

(35) If container j is the direct successor of container i considering ALV transfer task, then the start date of ALV transfer task of j is at least equal to the completion time of ALV transfer task of i added to the ALV routing time between the unloading position of ASC assigned to i and the QC associated with j.

3.2.3 NP-Hardness of IPLAVS - case of ALV

We use the same demonstration as in part 3.1 (case of MACT using straddle-carriers) with minor modifications which concern essentially waiting time (naturally nil in the case of MACT using ALV) and routing paths (for MACT using ALV, routing path are determined considering the associated traffic layout presented in chapter 2).

3.2.4 Evaluation of Lower-Bounds

For Lower-Bound evaluations, we consider all results of part 3.1 taking into account traffic layout particularities in the case of Handling System using ALV and especially the associated

routing paths.

3.3 Integrated problem of location assignment and AGV scheduling

For MACT using AGVs, we consider the same theoretical results of part 3.2 considering waiting time at ASC unloading position. NP-Completeness of IPLAVS (Integrated Problem of Location Assignment and Vehicles Scheduling) in the case MACT with AGV is proved using the same demonstration of part 3.1 treating the problem for handling system with Auto-Strad. The lonely difference between the two demonstrations is the fact that for the case of AGV it is necessary to consider a nil waiting time at ASC unloading position to construct the polynomial reduction; in other terms we consider instances with storage locations positioned at bay entries and negligible ASC unloading time.

For modelling, we consider constraints (1) to (37) of mathematical model in part 3.2 added to next constraint.

$$\forall (\boldsymbol{i}, \boldsymbol{p}) \in \boldsymbol{C} \times \boldsymbol{P}:\ \boldsymbol{t_2}(\boldsymbol{i}) \geq \boldsymbol{t_{ASC}}(\boldsymbol{i}) \qquad \textbf{(38)}$$

Constraint (38) insures interaction condition between AGV and ASC which is a particularity of handling system using these equipments. In fact, at the contrary of ALV, AGV has to wait for ASC to unload container and then it can move to transfer next container under QC.

For Lower-Bound evaluations, we consider all results of part 3.1 taking into account traffic layout particularities in the case of Handling System using ALV and especially the associated routing paths.

3.4 Mono-objective optimization of IPLAVS with new cooperative tabu search approach

Before solving IPLAVS considering its Multi-objective aspect, we solved it considering mono-objective version with the only objective of minimizing makespan. After testing different approaches, we opted to develop a new cooperative tabu search algorithm for an efficient resolution of mono-objective IPLAVS.

Cooperative approaches are applied to different meta-heuristics and especially tabu search, genetic algorithm and simulated annealing algorithm. In this part, we propose a cooperative exploration of the solution space based on tabu search, threading and communication. To test the efficiency of our approach in the general context of combinatorial optimization, the algorithm is applied to the Traveling Salesman Problem (TSP) which is known to be NP-Complete. To prove the efficiency of our approach, we compare it to classic tabu search and Late Acceptance Hill-Climbing algorithm (LAHC).

Late Acceptance Hill Climbing is a recent and efficient meta-heuristic in the realm of local search based algorithm. LAHC delay the comparison between neighbors of current solution and compare new candidates to solutions having been current for several iterations of the exploration process. The LAHC was successfully tested for exam timetabling PATAT 2008 conference in, the traveling salesman problem and the magic square problem. LAHC is known to be very efficient considering especially running time.

In the last 15 years, parallelization of TS has been treated by several studies. Threading and parallel computing offer different advantages to meta-heuristic approaches. These advantages concern essentially the sharing of computational tasks and the possibility of communication between different processes in order to improve the parameters of search. The main goal of meta-heuristic parallelization is to make shorter running time. However, parallelization can improve solution efficiency for approaches which don't surely converge to the global optima.

Three kinds of parallelization are identified in reviews of literature:

- ✓ **Operation parallelization**

It is a low-level parallelization sharing and accelerating computational tasks of algorithms.

- ✓ **Search space decomposition**

This approach use parallel searches of sub-solutions in different sub-spaces. Global solutions are regularly given by the set of sub-solutions.

✓ **Multi-search threads**

This approach uses several searchers (or threads) at the same time. The threads can be independent or cooperative.

We propose a new Multi-search threads approach which is a cooperative tabu search (CTS) based on threading and interaction between the different used threads. We apply this approach to NP-Complete combinatorial problem which is TSP. To prove the efficiency of our approach we compared the numerical results of CTS with those of classic TS and LAHC heuristic.

TS is an efficient approach of resolution for NP-hard combinatorial problem developed by Glover [1] in 1986. TS algorithm starts from an initial solution and move from neighborhood to neighborhood. At every neighborhood the best solution is selected. The efficiency of TS algorithm compared to others approaches is the direct consequence of its opportunist and intelligent exploration of the solution space. In fact, the exploration is regulated by a memory list named "tabu list" which contains the set of n last selected solutions. The tabu list is used to not end-up trapped in local optima. The main goal of meta-heuristic parallelization is to solve NP-hard problems in a small amount of time. In some situations, the parallelization improves the solution efficiency. Two level of parallelization are identified: a low level and a high-level. The low level parallelization is based on sharing computational tasks to accelerate the resolution. The high-level parallelization is a cooperative strategy using communication between different threads in order to improve the efficiency of search parameters and solve the problem in a small run time. Cooperative multi-search threads approaches are generally based on communication and cooperation between the different used threads. Different levels of communication and cooperation can be considered.

In the taxonomy of El-Abed and Kamel [2], two kinds of Cooperative multi-search threads approach are identified in review of literature: the heterogeneous approach and the homogeneous approach. The heterogeneous approaches use threads which apply different kind of algorithms in parallel with different possibilities of communication and cooperation. The homogeneous approaches use threads which apply the same algorithm in parallel, generally with communication

of integral information to improve the parameters of search. These methods are applied to different meta-heuristics and especially TS [3], genetic algorithm [4] and simulated annealing algorithm [5].

Our cooperative strategy is based on TS, threading and communication between the different used threads. Communication is exploited to regulate and improve the direction of exploration for each thread (or searcher). Considering the theoretical convergence of classic TS, the principal goal of our CTS strategy is to make shorter the run time.

For the description of our CTS approach, we consider combinatorial problems of minimization.

In our approach, we use cooperative exploration of the solution space. Many threads explore the solution space using the opportunist strategy of TS algorithm but with a common tabu list. The different threads communicate with each-other. This communication insures the use of the best current solution data to establish a set of actions and reactions for each thread during the solution space exploration. These actions and reactions are described in next part.

3.4.1 Concept of action and reaction

We construct the processes of actions and reactions as follow:

Action: When one thread finds a solution better than the last absolute best solution (the best solution considering all threads), it gives this solution to the others threads. The communicated solution becomes the current absolute best solution. We describe this process of the mechanism as an action.

Reaction: Our approach considers a tolerance of deviation between the current solution of every thread (excepting the thread given the current absolute best solution) and the current absolute best solution. We denote this limit of deviation by Δ. If the deviation between the current solution and the absolute best solution known at the current time becomes greater than Δ, the thread associated to that current solution modifies its search. This search modification is the reaction of restarting exploration from the (current) absolute best solution. We describe this process of the mechanism as a reaction.

For each thread, the exploration of solution space is composed of three processes: the interaction processes (action-reaction) discussed in the last part, and two others process which are the increase of deviation limit and the deterioration of current solution in order to find new effective exploration zones of the solution space (a zones of which contain one or many minima). The increase of deviation limit is insured by the dynamic evaluation of Δ described in next part.

3.4.2 Dynamic evaluation of deviation limit Δ

Consider the evaluation of Δ (the limit of deviation) during the exploration. The limit of deviation between the absolute best solution and the current solution of each thread is variable. Δ depends on the value of the absolute best solution which is minimized during the exploration.

$$\Delta = \frac{S_A}{R}$$

S_A: The value of A.B.S (Absolute Best Solution).

R: Variable integer dependent on the exploration step of the considered thread. In the algorithm, presented next, R is denoted by Th.R for each thread Th.

The value of R is also variable and depends on the variation of the current best solution (the current best solution of the thread and not the absolute best solution). R is initialized by R0 and when the variation of the current best solution becomes nil, R regresses (Δ increases).The regression of R (the increase of Δ) is compounded by k steps and at every step R is divided by a static value D (Δ is multiplied by D). This mechanism is a diversification strategy used to explore relatively bad solutions in order to find others optima.

For each used thread, at the end of this step, the variation of current best solution stay nil for significant run time even if we increase another time the limit of deviation Δ.To improve the solution space exploration we deteriorate the current solution in order to diversify the explored solutions and find more minima. This process is described in the following part.

3.4.3 Deterioration process

When the increase of Δ becomes inefficient to find more optima another mechanism of diversification is used. In fact, if after the k steps of increase of R, the variation of the local best solution becomes again nil, we apply a forced increase (gap regression) of the current solution until it reaches a sufficiently bad value γ . This value is evaluated as next:

$$\gamma = C + random\,(p) \times \frac{|\,U.B - C\,|}{p}$$

U.B: Upper Bound

C: The value of the current solution of the thread

p: a chosen integer

random(p): a random integer between p/2 and p.

3.4.4) Algorithm

In this part, we introduce the algorithm of our CTS approach. We consider next variables:

Th: a thread.

Th.current: The current solution of the thread Th.

Th.best: The best solution found by Th.

Absolute Best: The best solution considering all the threads.

Th.step: The step of resolution for the considered thread Th. For each thread, the process is composed of four steps. Steps 0 to 2 are characterized by the interaction between the considered thread and the other threads. During step 3, the current solution of the thread Th is deteriorated.

Th.negative iter: The number of successive negative iterations. We define a negative iteration as an iteration which has not improved the best solution of the thread Th (Th.best).

Th.degradation( ): The process of degradation of current solution of the thread Th (Th.current). During this process, the thread Th is isolated from the other threads.

Th.TS():The process of classic tabu search used after the degradation of the current solution of Th. During this process, the thread is partially isolated from the other threads, actions and reactions are not supported and only the absolute best solution is communicated.

Th.R (or R) and D are defined in part 3.3. 2.

R_0: Initial value of R.

We present next the pseudo-code of our strategic exploration considering the described variables.

CTS Algorithm

$for\ (Th\ \in list\ of\ threads)$

$Th.R\ \leftarrow\ R_0$

$While\ (cost > cost_{Max})$

$for\ (Th\ \in list\ of\ threads)$

{ $if\ (Th.step < 3)$

{ $Th.current\ \leftarrow best\ Neighbor\ out\ Tabu\ List$

$if\ (Th.current < Th.best)$

{ $Th.best\ \leftarrow Th.current$

$Th.negative\ iter\ \leftarrow 0$ }

$else\ \ Th.negativ\ iter\ \leftarrow Th.negative\ iter + 1$

$if\ (Th.negative\ iter > limit)$

{ $Th.step\ \leftarrow Th.step + 1$

$Th.R \leftarrow \frac{Th.R}{D}$

$Th.negative\ iter \leftarrow 0$ } }

$else\ if\ (Th.step = 3)$

{ $Th.degradation( )$

$While(Th.\text{best} > Absolute\ Best)$ $Th.TS( )$

$Th.step \leftarrow 0$

$Th.R \leftarrow R_0$ }

$if\ (Th.current < Absolute\ Best)$

$Absolute\ Best\ \leftarrow Th.current$

$else\ if\ \left(Th.current > Absolute\ Best + \frac{Absolute\ Best}{Th.R}\right)$

$Th.current\ \leftarrow Absolute\ Bes\text{t}$ }

Our approach uses a mechanism of interaction (actions - reactions) between the different searchers (or threads) to improve the solution space exploration. We applied our approach to the Traveling Salesman Problem (TSP).

In what follows, we compare our CTS approach to classic TS and Late Acceptance Hill-Climbing algorithm (LAHC).

3.4.5 Application to the traveling Salesman Problem

TSP is one of the most famous NP-Complete Combinatorial problems in literature. Many methods are applied to TSP. One of the most efficient approaches used to solve TSP in reasonable run time is the Late Acceptance Hill Climbing heuristic (LHAC) [6] [7]. Note that LAHC is a performed version of the greedy Hill-Climbing algorithm.

In this part, we show the efficiency of our CTS approach compared to the classical TS and LAHC algorithm for different instances of TSP.

If we consider initially a limited run time, the cooperation between the different threads improves considerably the efficiency of solutions given by classical TS. Considering the numerical results of TABLE 10, CTS is much more powerful than classical TS. LAHC is applied to TSP.

We used the java code of LAHC developed by the authors [6] [7]. We coded TS algorithm and our CTS approach also in java language. Simulations have been run on a PC with an INTEL Xeon running at 2.67 GHz under the Windows 7 operating system (32 bit). The CTS resolution is established with 12 threads. The end condition proposed for CTS is not considered in the presented simulations.

We compare our CTS approach and LAHC for instance of more than 50 cities and less than 144 cities. In Table I0 we show that for these instances our CTS approach is more efficient than LAHC considering solution gap. Note that for each solution S, the gap is evaluated as next:

gap(S) = (value (S) – Lower Bound) / Lower Bound).

Note that, for instance of more than 100 cities CTS need more computational capacity to solve efficiently TSP.

Resolution of TSP by CTS, TS and LHAC			
Instance	**TS V / R.T**	**CTS V / R.T**	**LHAC V / R.T**
Berlin 52	7876 / 1s to15 h	7542 (optimal) / 6 s	7570 / 2s
ST70	688 / 1s to 30 h	675 (optimal) / 19 min	681 / 5 s
Eil76	572 / 1s to 30 h	538 (optimal) / 21 min	548 / 2 s
KroC100	22144 / 1s to 60 h	20749 (optimal) / 6 h	20940 / 4s
Eil101	676 / 1s to 60 h	629 (optimal) / 6 h	639 / 4s
Pr144	62960 / 1s to 60 h	58537 (optimal) / 1h	58607 / 3s

V / R.T: Value / Run Time

Table 10

3.5 Multi-Objective Optimization

For many optimization problems, to take a decision, we have to satisfy different criteria. Considering the case of container terminal managed by straddle carriers, a trial objective is to minimize the makespan which is the date of the last task, but in a real situation, the decision has to take into account other criteria such as the sum of vehicle operating time, the number of straddle carriers used, the number of storage bays used etc. To solve the integrated problem of location assignment and straddle carrier scheduling in maritime terminal at import, we have two possible approaches. The first is to consider a known operating coast function which we evaluate considering the handling time, the equipment used and the final storage space configuration. The second is to solve the problem considering the strict multi-objective aspect.

Consider S a set of realizable solutions, n > 1, f_i ($1 \leq i \leq n$) a scalar function over S. Multi-objective optimization can be represented mathematically as:

$\min (f_1(x), f_2(x), \ldots, f_{n(x)})$

$x \in S$

When we solve a multi-objective problem, we cannot consider directly the ordinary scalar optimality. In fact, a Pareto optimality concept is defined:

- ✓ $\forall (x, y) \in S^2, x \neq y: x\ weakly\ pareto - dominates\ y\ if\ and\ only\ if$:

  $\forall\ i\ \in \{1,2..,n\}: f_i(x) \leq f_i(y)$

  $\exists\ i\ \in \{1,2,..,n\}: f_i(x) < f_i(y)$

- ✓ $\forall (x, y) \in S^2, x \neq y: x\ strongly\ pareto - dominates\ y\ if\ and\ only\ if$:
  $\forall\ i\ \in \{1,2..,n\}: f_i(x) < f_i(y)$

- ✓ A solution x is Pareto-optimal if and only if there does not exist another solution y which weakly Pareto-dominates x.

- ✓ A solution x is weakly Pareto-optimal if and only if there does not exist another solution y which strongly Pareto-dominates x.

3.6 Multi-Objective Tabu Search Algorithm – MOTSA

3.6.1 Data for solution representation

For the most general context of IPLASS, each solution is a four-dimensional vector and every variable of this vector comprises four integers which represent a task, a container, a vehicle and a storage location. With our traffic layout particularities (at every time the vehicle to use for the next container transfer is the first straddle carrier returning to quay entry) and established container transfer schedule, the decisional problem concerns only the choice of storage places for every container. We can establish a total container order without changing makespan optimality for general real configuration of storage space. We conclude that we can use a data list to represent solutions. Consider an instance of 10 containers and 30 storage places. We denote by ((1, 10), (2,

13), (3, 3), (4, 1), (5, 0), (6, 6), (7, 7), (8, 22), (9, 29), (10, 5)) the solution S which assign for containers 1 to 10 the respective storage locations 10, 13, 3, 1 etc. Considering the established container order, S can be also represented by the list (10, 13, 3, 1, 0, 6, 7, 22, 29, 5).

3.6.2 Approach description

To solve efficiently IPLAVS, we developed a Multi-Objective Tabu Search Algorithm (MOTSA). Our resolution is a dynamic opportunist exploration of the solution space considering a specific neighborhood. We defined the neighborhood taking into account the different goals to optimize. The exploration of solution space is composed of different cycles and every cycle is composed of different periods.

The initial solution is elected from a sufficiently big set of feasible solutions.

Considering a current solution, the best neighbor is determined using weighted sum method. In fact, we evaluate linear objective function updated at the beginning of every cycle and at the beginning of each period. The elected neighbor is added to a Tabu list and declared Tabu during the current period. At the end of current period, the Tabu list of best neighbors will be cleared.

Consider now every current solution (at the beginning the current solution is the initial solution). Firstly, if the resolution process is at the beginning of a period, objective weights are updated. Then we elect the best neighbor considering the new objective coefficient update. If the elected neighbor is not dominated by the Pareto solutions, we add it to the set of Pareto solutions and we update the Pareto list.

Because the problem is a mixed-integer problem, not only the best neighbor is considered when we update Pareto list, but also all non-dominated solutions in every neighborhood to approach more efficiently the non-convex Pareto Front.

Every non-dominated solution in the current neighborhood is added to the Pareto list. After every solution injection, the Pareto list is updated and each dominated solution is deleted from the list.

For next step, the neighborhood of best current solution is explored.

When the current Pareto-optimal solutions satisfy the user, the procedure is stopped and user has to choose a solution from the Pareto list.

3.6.3 Neighborhood construction

The neighborhood is composed of three different sub-neighborhoods: neighborhood considering storage location aspect, neighborhood considering storage bay aspect and neighborhood considering the number of straddle carriers

3.6.3.1 Neighborhood considering storage location aspect

For every solution, the global neighborhood contains |C|*(|P|-1) elements. Considering the real dimensions of instances, it is not effective to select all the solutions during the exploration of each neighborhood. Only a sufficient random part of the neighborhood is considered. Consider a solution S = (P1, P2, …, P|C|), where Pi is an element of P for each integer i between 1 and |C|. Consider V(S) the neighborhood of S, then:

$$\forall N \in V(S), N = (N_1, \dots, N_{|C|}) : \exists! i \leq |C|, N_i \neq P_i$$

In figure 16, we present two possible neighbors for solution S considering storage location aspect.

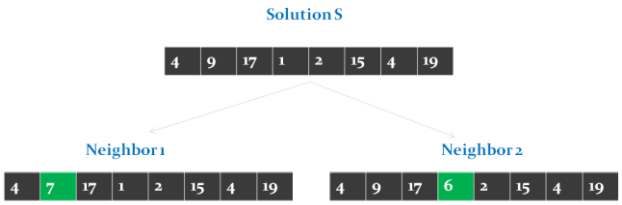


Figure 16 - Neighborhood considering storage location aspect

3.6.3.2 Neighborhood considering storage bay aspect

Consider now a solution S, B(S) the set of bays used considering solution S and B(Si) the bay of storage location assigned to container number i.

$$N \in V^+ \Leftrightarrow B(S) \subset B(N), |B(N)| = |B(S)| + 1 \; and \; for \; each \; i \leq |C| \; if \; B(N_i) \in B(S) \; then \; N_i = S_i \; else \; N_i \neq S_i$$

$N \in V^- \Leftrightarrow B(N) \subset B(S), |B(N)| = |B(S)| - 1 \; and \; for \; each \; i \leq |C| \; if \; B(S_i) \in B(N) then$
$N_i = S_i \; else \; N_i \neq S_i$

In figure 17, we present two possible neighbors of a giving solution S.

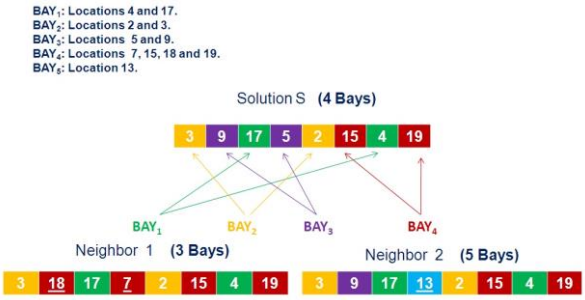


Figure 17 - Neighborhood considering storage bay aspect

3.6.3.3 Neighborhood considering straddle carriers used

Considering solution S using n vehicles, two neighbors are possible: one with n+1 (if n is smaller than maximal number of straddle carriers used) vehicles and another with n-1 vehicles (if n is larger than 1).

3.6.4 Initial solution election

The initial solution is elected considering a set of n solutions. The size of this set depends on instance size and exactly on the number of containers.

3.6.5 Cycles and periods

The exploration of solution space is composed of different cycles and every cycle is composed of different periods. Each period is composed of a set of neighborhood explorations.

3.6.6 Objective weights and distant elements

The MOTSA is based on a cyclic opportunist exploration. The updated objective weights determine the influence of every objective at every exploration period. For every objective k, the weight W_k, is evaluated as follows:

$$W_k = \alpha_k \times C_k$$

α_k: The α -weight of objective k. The α -weights are updated at every new period. To update α -weights, we use weighted sum method explained in part 3.4.8.

C_k: The cost of objective k during the current cycle. C_k has to be sufficiently large compared to the α -weights. We use dynamic cost C_k in order to perform the classic weighted sum method. In fact, we regulate the exploration of Pareto Front Region (PFR) with strategy of compromise and with diversification of the value of each objective unity (the value of losing or earn one unity of handling times or vehicles used or other objective).

At the beginning of every period, we select a neighbor which represents a maximal distance considering the Pareto List. We can describe this neighbor as a distant element. We use next function for that selection:

$distantElement(V(S), L)$: From the neighborhood of solution S, we select the solution which maximizes next distance d evaluated as:

$\forall v \in V, d(v) = \sum_{w \in L} |F_k(v) - F_k(w)|/h_k$. Where $h_k$is the difference between the maximal and the minimal values of objective k considering the current solutions of Pareto list.

3.6.7 Costs update

Every cycle is composed of a set of periods. At the beginning of each cycle, the cost of every objective is updated. To update the different costs we consider two factors; the first is the deviation of the objectives from their lower bounds considering the solutions in the current Pareto List; the second is the values of objective costs during the precedent cycles. We describe the cost of each objective k as an objective cost and we denote by it c_k.

For the first factor influencing the update of objective cost c_k, we consider all the elements of Pareto List and we define the average gap of objective k $\widetilde{gap}_k$as follows:

$$\widetilde{gap}_k = \frac{\sum_{S \in PL} gap_k(S)}{|PL|}$$

Where |PL| is the number of solutions in Pareto List PL and $gap_k(S)$ the gap of solution S considering the objective k.

We evaluate this gap as follows:

$$gap_k(S) = \frac{F_k(S) - LB_k}{LB_k}$$

$F_k(S)$: *The value of objective k considering solution S.*

LB_k: *The lower bound of objective k.*

After the evaluation of the average gaps of Pareto List PL, we store them in a list L^G in ascending order and we define the first priority of objective k as follows:

$$priority_k^1 = index\ of\ \widetilde{gap}_k\ in\ L^G$$

For the second factor influencing the update of objective cost c_k, we consider all precedent values of c_k for each objective k. We evaluate the average cost of objective k (noted $\tilde{c}_k$) as follows:

$$\tilde{c}_k = \frac{\sum_{u \in U} c_k^u}{|U|}$$

U: The set of precedent cycles

c_k^u: The cost of objective k during the cycle u.

After the evaluation of the average cost of every objective k (AC_k) during the precedent cycles, we store them in a list L^U in descending order and we define the second priority of objective k ($priority_k^2$) as follows:

$$priority_k^2 = index\ of\ AC_k\ in\ L^U$$

The new cost of each objective k is evaluated as follows:

$$c_k = 10^{2 \times priority_k^1 + priority_k^2}$$

3.6.8 α-weights updat

At the beginning of each cycle, the cost of every objective is updated, then these costs are fixed and no other update is possible until the end of the cycle. However, during the cycle, weighted sum method is applied to more diversify the exploration of Pareto Front Region (PFR).

During each cycle, at the beginning of every period, the □-weights are updated. Update process is based on two considerations; the $\alpha - weight$ history during current cycle and the deviation of the objectives from their lower bound during previous periods in current cycle.

For each objective, the new updated $\alpha - weight$ is a perturbation of its last □-weight. Perturbation can be positive or negative with a probability which depends on the average gap ($\widetilde{gap}$) and the average $\alpha - weight$ ($\tilde{\alpha}$) of the objective.

Average gap is evaluated as in part 3.4.7. Average □-weigh is evaluated for each objective k as follows:

$$\tilde{\alpha}_k^{c,p} = \frac{\sum_{m \in pred_p^c} \alpha_k^{c,m}}{\left|pred_p^c\right|}$$

c: current cycle

p: current period in cycle c.

$pred_p^c$: $the\ set\ of\ periods\ preceeding\ period\ p\ during\ cycle\ c.$

$\left|pred_p^c\right| = the\ size\ of\ pred_p^c.$

$\tilde{\alpha}_k^{c,p}$: $the\ average\ weight\ of\ objective\ k\ considering\ pred_p^c.$

$\alpha_k^{c,m}$: $the\ \alpha - weight\ of\ objective\ k\ during\ period\ m\ of\ cycle\ c.$

After the evaluation of the average weight of every objective k during the periods preceding period p in current cycle c, we store them in a list L^W in descending order and we define the third priority of objective k ($priority_k^3$) as follows:

$$priority_k^3 = index\ of\ \tilde{\alpha}_k^{c,p}\ in\ L^W$$

At the beginning of period p, the probability of positive deviation of the weight associated to objective k,Prb_k^p, is evaluated as next:

$$Prb_k^p = \frac{2 \times priority_k^1 + priority_k^3}{3 \times |O|}$$

$|O|$: $the\ number\ of\ objectives\ to\ optimize.$

For the objective having favorable average Gap (AG) compared to the other objectives, the probability of negative perturbation of the associated weigh is greater especially if the average value of the associated weigh is significant. On the contrary, the probability of positive deviation is greater.

After each update, the vector of new $\alpha - weights$ ($\alpha = (\alpha_1, \dots, \alpha_{|O|})$) is added to the Tabu list of weight used. When the Tabu list size limit is reached, as for classical Tabu list, the oldest vector of weight is removed.

At the end of every cycle, the Tabu list of $\alpha - weight$ vector is initialized and all weight vectors are removed from the Tabu list.

We present next the algorithm of the described process of weight update.

Δ: $the\ absolute\ value\ of\ the\ sum\ of\ negative\ perturbations$

Q: $the\ number\ of\ positive\ perturbations$

p: $the\ current\ period.$

$\sigma_k^p = the\ sign\ of\ the\ perturbation\ of\ objective\ k\ during\ period\ p.$

α^p: $the\ vector\ of\ objective\ weights\ in\ period\ p.\ \alpha^p = (\alpha_1^p, \alpha_2^p, \dots, \alpha_k^p, \dots, \alpha_N^p)$

L_W: $The\ Tabu\ list\ of\ used\ vectors\ of\ weigtht.$

$for\ (k \leftarrow 1\ to\ k = N)$

{ $\delta \leftarrow random\ real\ in\ [0; 1]$

$if\ (\delta < Prb_k^p)$

{ $\alpha_k^p \leftarrow \alpha_k^{p-1} - \varepsilon$

$\sigma_k^p \leftarrow -1$

$\Delta \leftarrow \Delta + \varepsilon$ }

$else$

{ $\sigma_k^p \leftarrow +1$

$Q \leftarrow Q + 1$ } } $//end\ for$

$for\ (k \leftarrow 1\ to\ k = N)$

$if\ (\ \sigma_k^p = +1)\ \ \alpha_k^p \leftarrow \alpha_k^{p-1} + \frac{\Delta}{N}$

$L_W \leftarrow L_W \cup \{\alpha^p\}$

$L_W \leftarrow L_W \setminus oldest\ vector\ (L_W)$

Note that "$oldest\ vector(L_W)$" is a function giving the oldest element injected to the list of weights L_W.

3.6.9 Pareto list update

After every election of a new current solution S, if S is not dominated by Pareto list elements, it is added to the Pareto list and Pareto solutions dominated by S are removed from the list.

3.6.10 Finalization condition

We consider the number of successive periods without change in the Pareto list (successive negative periods). At any time in the resolution process, if this number is bigger than a chosen number M_{Max}, the exploration is stopped. Another condition of finalization is when the number of

exploration periods is equal to a chosen number E_{Max}. For experiments, we change the second condition with a maximal run time. The user can also stop the process if he is satisfied by the set of Pareto solutions.

3.6.11 MOTSA

M: Pareto list evolution indicator considering period iterations. MOTSA use M to determine the end of solutions space exploration.

N: Pareto list evolution indicator considering exploration iterations. MOTSA use N to determine a good timing to regulate objectives coefficients.

S: Current solution. At every iteration of the exploration process, the current solution is the elected neighbor of the last solution.

Elite Neighbor (S): Function returning an elected neighbor which is the best neighbor considering the current solution S and the current objective coefficient regulation.

L: Pareto list, which contains the different Pareto solutions. MOTSA updates this list at every exploration of a new Pareto solution (positive exploration).

M_{max}, N_{max}, E_{max}: Maximal tolerable values of M, N and E.

MOTSA uses two kinds of iteration: period iterations and exploration iterations.

A negative exploration is an exploration which doesn't affect the Pareto list while a positive exploration is an exploration which affects the Pareto list.

P: Period, which is a set of exploration. The end of a period is when the number of successive negative explorations is equal to N_{max}.

Negative period is a period which doesn't affect the Pareto list while a positive period is a period which affects the Pareto list.

The end of MOTSA resolution process is when the number of successive negative periods is equal to M_{max}.

H(x): the promised indicator of solution x.

H(x) is evaluated as next: $for\ (x \in L)\ if\ x\ equal\ S \quad H(S) \leftarrow x.h$

$H_{promised}$: Characterization of promised solutions. This value is used to establish the aspiration strategy.

$elite\ neighbor(S)$: The best neighbor of S considering the last update of weights. The elected neighbor can be a Tabu solution with a small probability.

V(S): Partial neighborhood of S.

V_H (S): a partial neighborhood of S with a size depending on H(S):

if $H(S) \geq H_{promised}$, then $|V_H(S)| = 2\ |V(S)|$, else $V_H(S) = V(S)$.

Algorithm

$P \leftarrow 0$, $M \leftarrow 0$, $E \leftarrow 0$, $S \leftarrow initial\ solution$, $L \leftarrow \{S\}$

initialize objective costs

While ((M <M_{max}) and (E<E_{max}))

{

Initialize α-weights

$N \leftarrow 0$

$negative\ period \leftarrow true$

if (new cycle) update objective costs

if (new period)

{ update α-weights

$S \leftarrow distantElement(V(S), L)$ }

While (N < N_{max})

{

$E \leftarrow E + 1$

$negative\ exploration \leftarrow true$

if (S is not in L) and S (is not dominated by the solutions in L)

{ $H \leftarrow 0$

$for\ x \in L$

if (x is dominated by S)

{ $L \leftarrow L \backslash \{x\}$

$H \leftarrow H + 1$ } // end if

$L \leftarrow L \cup \{S\}$

$x.h \leftarrow H$ } // end if // x.h is an efficiency indicator of solution x

$if\ (S\ is\ not\ in\ tabu\ list)\ \ or\ \left(H(S) \geq H_{promised}\right)$

{ $if\ (S\ is\ not\ in\ tabu\ list)\quad T \leftarrow T \cup \{S\}$

$for\ v \in V_H(S)$

if (v is not dominated by solutions of L)

{ $N \leftarrow 0$

$M \leftarrow 0$

$if\ (negative\ exploration = true)$

$negative\ exploration \leftarrow false$

$if\ (negative\ period = true\ )$

$negative\ period \leftarrow false$

$for\ x \in L$

if (x is dominated by v)

$L \leftarrow L \backslash \{x\}$

$L \leftarrow L \cup \{v\}$ } // end for

if negative exploration $N \leftarrow N + 1$

$S \leftarrow elite\ neighbor(S)$

} // end if

$P \leftarrow P + 1$

if (negative period = true) $M \leftarrow M + 1$ } // end

3.6.12 Solution election

Considering the Pareto list elements, we elect a set of solutions using efficiency indicators EI. The first indicator of efficiency is evaluated as follows:

$$EI(S) = \sum_{k \in O} \sum_{p \in PL} I_{S,p,k}$$

Where, $I_{S,p,k} = 1\ if\ F_k(S) \leq F_k(p) - \delta_k, 0$ else.

$\delta_k is\ constaint\ depending\ on\ the\ smaller\ significant\ variation\ in\ objective\ k.$

For experiments, the value of $\delta_k$depends on the objective. For the first, fifth and sixth objectives (objectives evaluating operating times): $\delta_1 = \delta_5 = \delta_6 = 60\ seconds$.

For the second, the third, the fourth, the seventh and the eighth objectives (objectives evaluating the number of equipment used and the location assignment cost): $\delta_2 = \delta_3 = \delta_4 = \delta_8 = \delta_7 = 0$ unity.

The second indicator of solution efficiency, noted as Ind_2, is equal to the sum of the orders of preference of the solution considering the different objectives. Consider solution S with the respective orders of preference I_1 until I_8 considering the eight objectives to minimize, the second

indicator of efficiency is equal to the sum of these orders: Ind_2 (S) = $\sum_{1\leq k\leq 8} I_k(S)$. The best solution considering the second indicator of efficiency is the one which minimizes it.

The third one, noted as Ind_3, is a linear function with choosing objective weights. The best solution for the third indicator is the one which minimize that chosen function. To choose an efficient weights, we have to consider the difference between the objectives in their intervals of variations. For example we can't accept a big number of unproductive moves to win only some seconds of operating time. In fact, the chosen weights have to establish some equilibrium between the different objectives.

The fourth indicator of efficiency, noted as Ind_4, is a function of the three first indicators. Consider a solution S and O_1, O_2 and O_3 the respective orders of efficiency of S considering all solutions in approximated Pareto-Front for the first, the second and the third indicators. The value of the fourth indicators is equal to the sum of these orders of preference: Ind_4 (S) = O_1 + O_2 + O_3.The best solution considering the fourth indicator is the one which minimize it.

The first and the second indicators of efficiency are theatrically adapted to the combinatorial aspect of the problem as a non-convex problem with a non-convex Pareto-Front Region. The third indicator of efficiency is easy to be adapted by operators in the container terminal considering the current situation and especially the current priorities. The fourth indicator of efficiency is able to give a solution taking into consideration the combinatorial aspect of the problem and the preferences of the operator.

To elect a solution from the Pareto List, we used 2D projections of the multi-objective space and we select one of the solutions proposed by the different indicators of efficiency.

3.7 Numerical experiments

3.7.1 Numerical experiments for Mono-Objective IPLAVS

In next table, we present numerical result, of CTS approach considering different instance sizes. Instances used for the different kind of MACT are equivalents.

To solve the problem considering mono-objective aspect, we do not tolerate any unproductive move

using additional hard constraint. For the last instance, we stopped the resolutions after 2 h of running time.

Numerical results for Mono-objective IPLAVS

Containers / Capacity of free storage space	MACT with ALV	MACT with AGV	MACT with Auto-Strad
100/1000	Gap: 0.0 Run Time: 30 s	Gap: 0.0 Run Time: 80 s	Gap: 0.0 Run Time: 77 s
200/2000	Gap: 0.0 Run Time: 120 s	Gap: 0.0 Run Time: 308 s	Gap: 0.0 Run Time: 135 s
500/5000	Gap: 0.0 Run Time: 702 s	Gap: 0.0 Run Time: 609 s	Gap: 0.0 Run Time: 550 s
1000/10000	Gap: 0.08 Run Time: 2 h	Gap: 0.12 Run Time: 2h	Gap: 0.15 Run Time: 2h

Table 11

3.7.2 Numerical experiments for Multi-Objective IPLAVS

We solve IPLAVS in MACT using our MOTSA and considering the three presented kinds of terminals: MACT with Auto-Strads, MACT with AGVs and MACT with ALVs. For the case of Auto-Strad Terminal, we generate realistic data considering the MCT “Terminal de Normandie” in Maritime Port of Le Havre in France. We modified the maximal capacity of storage locations (these of "Terminal de Normandie") in order to treat large instances of the problem. For the case of MACT with ALVs and MACT with AGVs, we generate similar instances based on the layout of “Terminal de Normandie” and taking into account the specificity of these cases; Fictive installation of ASCs is considered and vehicle routing paths are adapted to each kind of terminal.

For each type of terminal, we show elected solutions given by MOTSA for a large instance of 1000 containers and a stacking space with a total storage capacity supporting 10000 containers. Only four elected solutions are presented for each case of equipment. 2D-projections of Pareto-Front are introduced and elected solutions are distinguished in these projections. The next computational results are obtained after 5000 neighborhood explorations. Lower bounds are determined to evaluate resolution quality.

3.7.2.1 Multi-objective resolution Case of MACT with Auto-Strads

For this instance, we evaluated the following lower bounds;

Makespan Lower Bound: 27110 sec

Lower Bound of |ASC*|: 11 ASCs

Lower Bound of $|P^*|$ (the number of storage location to be used): 112 locations; Lower Bound of |V| (the number of Auto-Strads to be used, considering an optimal makespan): 18

Lower Bound of $\sum_{b\in B^*} t_b$(total storage bay occupation time): 14000 s

Lower Bound of $\sum_{v\in V} t_v$(total Auto-Strad routing time): 465465 s

Lower Bound of LAC 1 (First Location Assignment Cost): 1 s

Lower Bound of LAC 2 (Second Location Assignemnt Cost): 0 UM (UM: Unproductive Moves).

To elect solutions from the set of approximated Pareto-Front, indicators of efficiency are presented in part 3.6.12. The projections of solutions proposed by indicators of efficiency are presented in the figures of 2D-projections using specific colors. In the following figures, we show the 2D-projections of more than 5000 solutions which approximate the Pareto Front.

2D-Projection of Pareto-Front Approximation

Plane of

Total Bay Occupation Time - TBOT

(Unity of graphic representation: 2000 sec)

AND

Total Straddle Carriers Routing Time - TSRT

(Unity of graphic representation: 1000 sec)

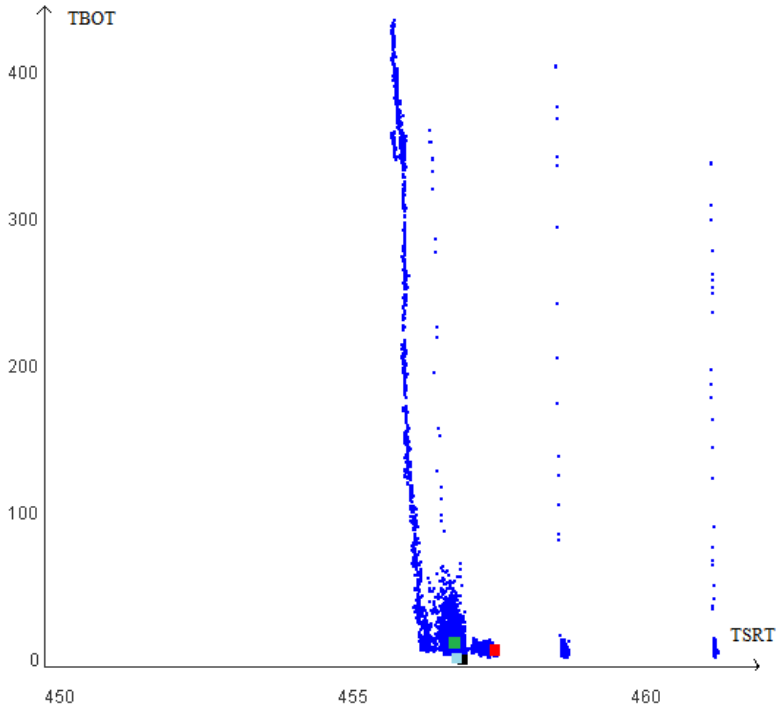


Figure 18

2D-Projection of Pareto-Front Approximation

Plane of

First Location Assignment Cost - LAC 1

(Unity of graphic representation: seconds of routing time

between two container locations assigned to the same client)

AND

Second Location Assignment Cost - LAC 2

(Unity of graphic representation: number of unproductive moves

to be caused by location decision)

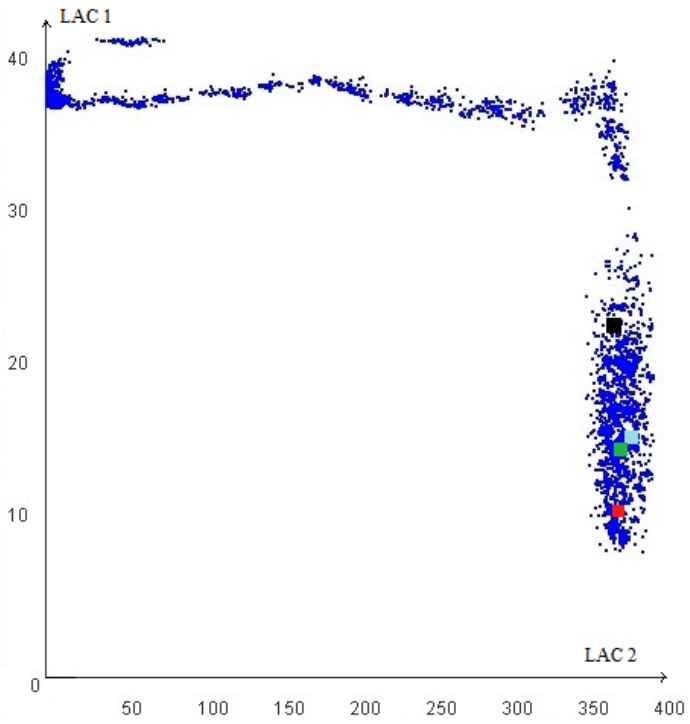


Figure 19

2D-Projection of Pareto-Front Approximation

Plane of

Makespan - C_{max}

(Unity of graphic representation: seconds of

deviation from Lower Bound)

AND

First Location Assignment Cost - LAC 1

(Unity of graphic representation: seconds of routing time

between two container locations assigned to the same client)

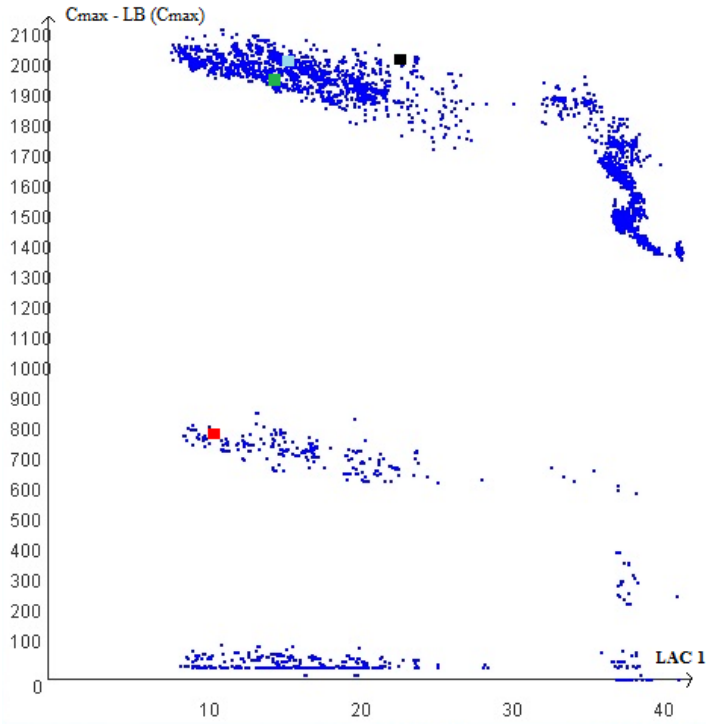


Figure 20

2D-Projection of Pareto-Front Approximation

Plane of

Makespan - C_{max}

(Unity of graphic representation: second of

deviation from Lower Bound)

AND

Number of storage bays to be used - |B*|

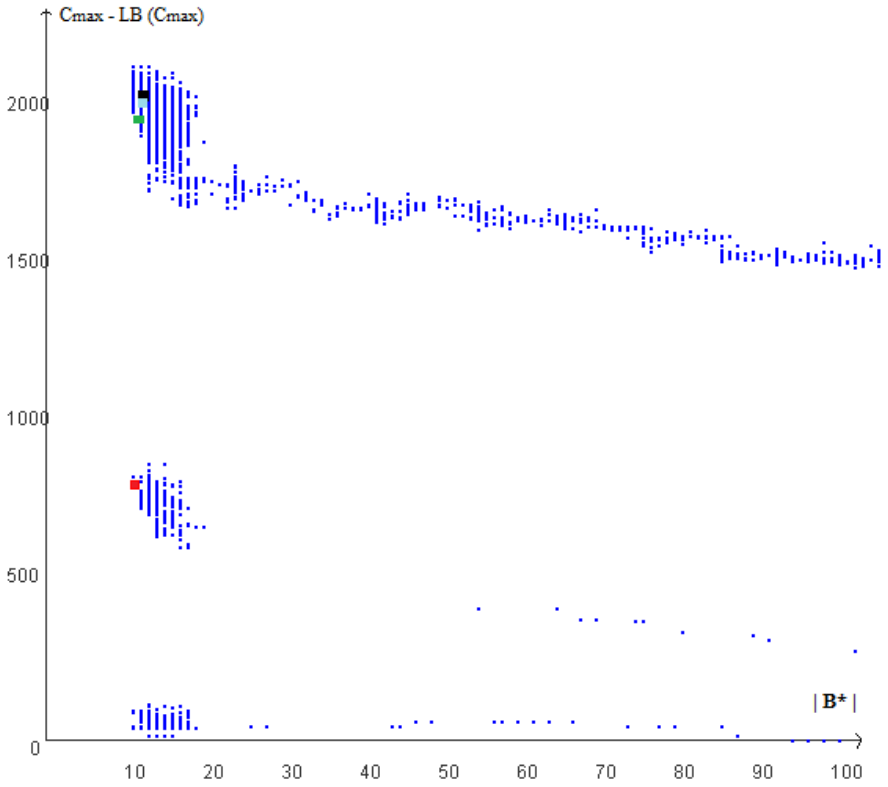


Figure 21

For 2D-projections showed in figures 18, 19, 20 and 21 and these presented in next sub-sections, we consider next representations of objective value projections:

Solution proposed by indicator 1
Solution proposed by indicator 2
Solution proposed by indicator 3
Solution proposed by indicator 4

In the following table, we present the best and worst values for which the gap reaches at every objective taking into account all the elements of approximated Pareto-Front.

Best and Worst Gap of approximated Pareto-Front

	C_{Max}	|V|	$\|B^*\|$	$\|P^*\|$	$\sum_{v\in V} t_v$	$\sum_{b\in B} t_b$	LAC_1	LAC_2
Best gap	0	0	0	0.21	0.09	0.02	7.8	0
Worst gap	0.08	0.17	14.09	7.43	0.21	62.06	40.93	399

Table 12

Solutions elected by indicators have objective values and objective gaps presented in next tables:

Objective values of elected solutions

	IND	C_{Max}	|V|	$\|B^*\|$	$\|P^*\|$	$\sum_{v\in V} t_v$	$\sum_{b\in B} t_b$	LAC_1	LAC_2
Solution 1	4616	27930	19	11	141	526144	25762	10.88	374
Solution 2	4371	29096	18	12	140	519467	30130	14.88	376
Solution 3	5012	29155	18	12	143	520205	14612	15.62	382
Solution 4	2651	29163	18	12	140	520523	15120	23.02	371

Table 13

Objective gaps of elected solutions

	IND	C_{Max}	|V|	$\|B^*\|$	$\|P^*\|$	$\sum_{v\in V} t_v$	$\sum_{b\in B} t_b$	LAC_1	LAC_2
Solution 1	4616	0.03	0.06	0	0.26	0.13	0.85	9	374
Solution 2	4371	0.07	0	0.09	0.25	0.12	1.17	13	376
Solution 3	5012	0.08	0	0.09	0.28	0.12	0.06	14	382
Solution 4	2651	0.08	0	0.09	0.25	0.12	0.06	23	371

Table 14

Solution 1: Solution proposed by first indicator.

Solution 2: Solution proposed by second indicator.

Solution 3: Solution proposed by third indicator.

Solution 4: Solution proposed by fourth indicator.

IND: Index of the solution in Pareto-List which represent the approximate Pareto-Front. All the elements of Pareto-List are presented is annexed parts.

We evaluate the gap of every objective value for each solution proposed by indicators of efficiency. The gap is equal to the difference between objective value and its lower bound divided by the lower bound.

$gap_k(S) = \frac{F_k(S) - LB_k}{LB_k}$ if k < 8

$gap_8(S) = F_8(S)$ (different evaluation because $LB_8 = 0$).

$gap_k(S)$: Gap of solution S considering only objective k.

$F_k(S)$: Objective value of solution S considering objective k.

LB_k: Lower Bound of objective k.

3.7.2.2 Numerical experiments for Multi-Objective IPLAVS (case of AGVs)

For this instance, we evaluated the following lower bounds;

Makespan Lower Bound: 27051 sec

Lower Bound of |ASC*|: 11 ASCs

Lower Bound of $|P^*|$ (the number of storage location to be used): 111 locations Lower Bound of |AGV| (the minimal number of AGV to be used, considering an optimal makespan): 16

Lower Bound of $\sum_{c \in ASC^*} t_c$(total ASC handling time): 78842 s

Lower Bound of $\sum_{v \in AGV} t_v$(total AGV routing time): 406406 s

Lower Bound of LAC 1 (First Location Assignment Cost): 1 s

Lower Bound of LAC 2 (Second Location Assignment Cost): 0 UM (UM: Unproductive Moves).

We present next the same 2D-Projections as in last section where solutions proposed by efficiency indicators are presented with the same colors. For the first location assignment (LAC_1) cost we consider for the unity of graphic representation: one second of routing time between two containers of same client. For the second location assignment (LAC_2) cost we consider for the unity of graphic representation: one unproductive move. Finally, for Equipment handling time we consider 1000 seconds as unity of representation. The number of equipment is represented with one unity.

2D-Projection of Pareto-Front Approximation

Plane of

Second Location Assignment Cost - LAC_2

AND

First Location Assignment Cost – LAC_1

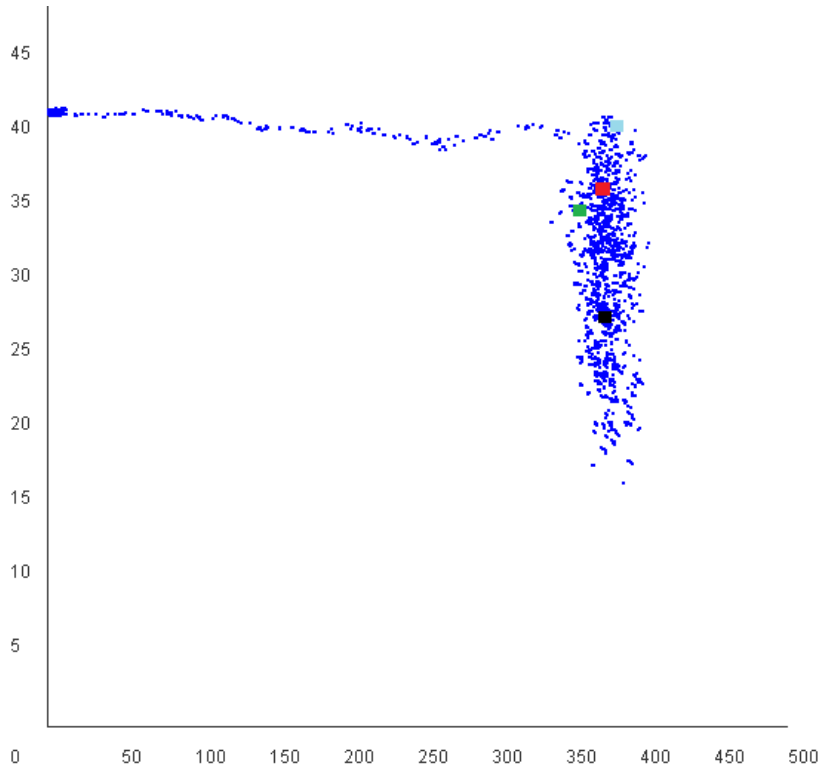


Figure 22

2D-Projection of Pareto-Front Approximation

Plane of

Number of locations to be used - $|P^*|$

AND

First Location Assignment Cost – LAC_1

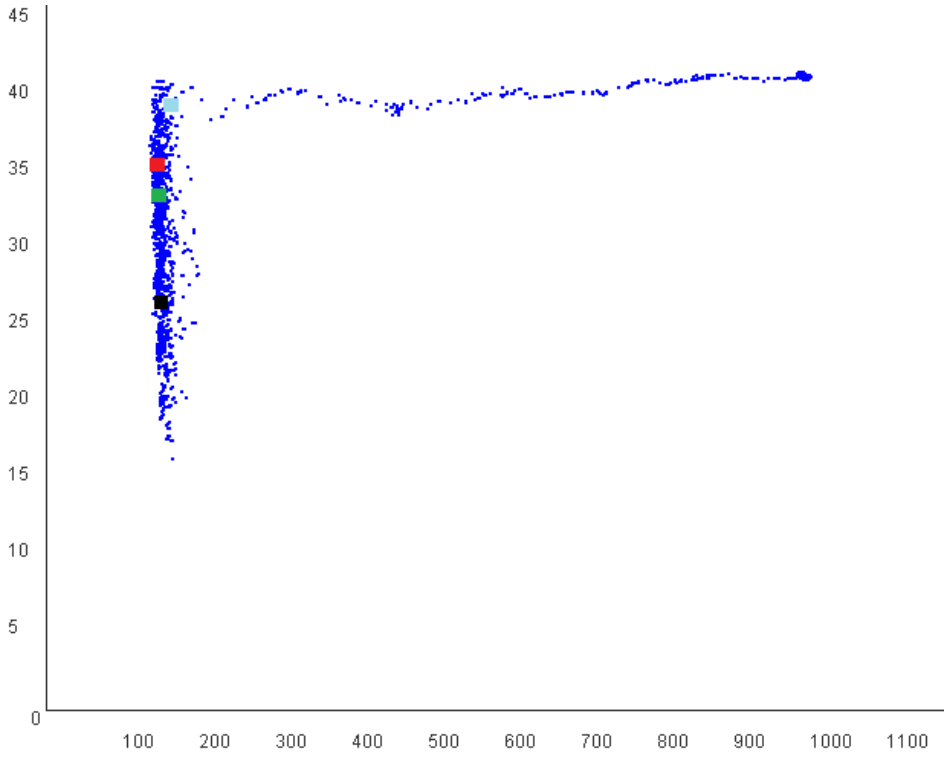


Figure 23

2D-Projection of Pareto-Front Approximation

Plane of

Number of ASCs to be used - $|B^*|$

AND

First Location Assignment Cost – LAC_1

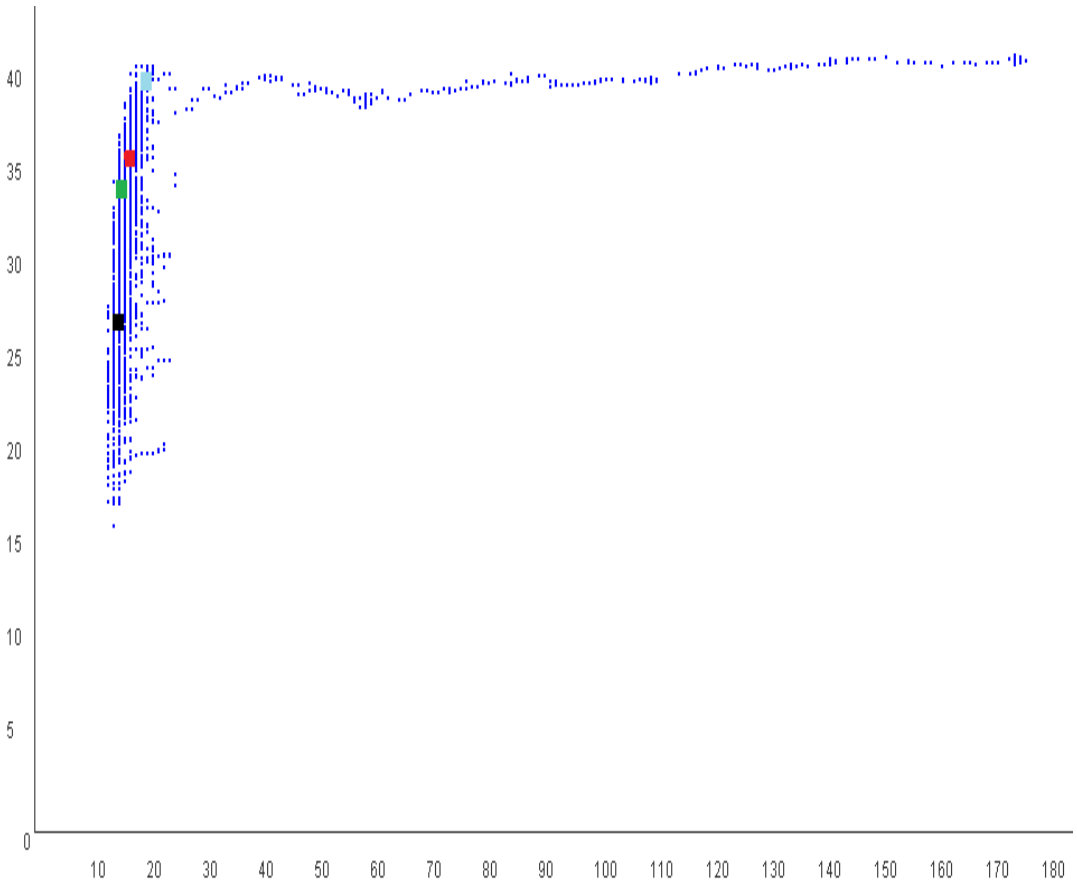


Figure 24

2D-Projection of Pareto-Front Approximation

Plane of

Total ASC handling Time

AND

First Location Assignment Cost – LAC_1

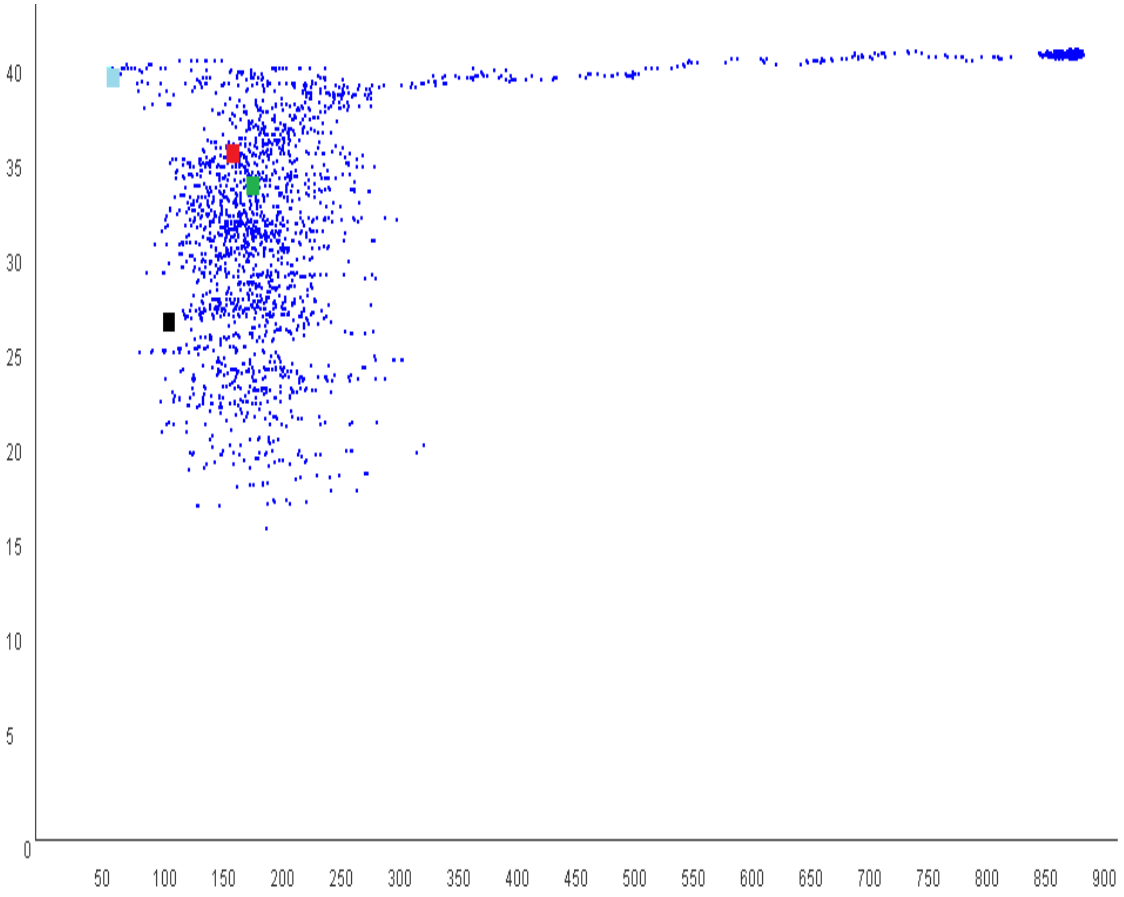


Figure 25

In next tables, we present the same elements as in last sub-section.

Best and Worst Gap of approximated Pareto-Front

	C_{Max}	|V|	|ASC*|	|P*|	$\sum_{v \in AGV} t_v$	$\sum_{c \in ASC} t_c$	LAC_1	LAC_2
Best gap	0	0	0.181	0.225	0.072	0.03	17	0
Worst gap	0.099	0.312	15	7.981	0.386	12.09	40	412

Table 15

Objective values of elected solutions

	IND	C_{Max}	|V|	|ASC*|	|P*|	$\sum_{v \in AGV} t_v$	$\sum_{c \in ASC} t_c$	LAC_1	LAC_2
Solution 1	1146	28403	16	16	148	450143	136585	28	369
Solution 2	1141	27532	17	14	146	463779	125557	28	370
Solution 3	246	27577	17	16	154	465518	81282	35	367
Solution 4	1119	27534	17	13	145	463772	113477	24	381

Table 16

Objective gap of elected solutions

	IND	C_{Max}	|V|	|ASC*|	|P*|	$\sum_{v \in AGV} t_v$	$\sum_{c \in ASC} t_c$	LAC_1	LAC_2
Solution 1	1146	0.05	0	0.45	0.33	0.1	1.6	27	369
Solution 2	1141	0.02	0.06	0.27	0.32	0.14	1.38	27	370
Solution 3	246	0.07	0	0.36	0.24	0.08	1.55	25	373
Solution 4	1119	0.06	0	0.45	0.34	0.07	1.17	25	379

Table 17

Solution 1: Solution proposed by first indicator.

Solution 2: Solution proposed by second indicator.

Solution 3: Solution proposed by third indicator.

Solution 4: Solution proposed by fourth indicator.

IND: Index of the solution in Pareto-List which represent the approximate Pareto-Front.

We evaluate the gap of every objective value for each solution proposed by indicators of efficiency as described in preceding part.

3.7.2.3 Numerical experiments for Multi-Objective IPLAVS (case of ALVs)

For this instance we evaluated the following lower bounds;

Makespan Lower Bound: 20013 sec

Lower Bound of |ASC*|: 11 ASCs

Lower Bound of |P*| (the number of storage location to be used): 112 locations; Lower Bound of |ALV| (the minimal number of AGVs to be used, considering an optimal makespan): 18

Lower Bound of $\sum_{c\in ASC^*} t_{ASC}$(total ASC handling time): 37156 s

Lower Bound of $\sum_{v\in ALV} t_v$(total ALV routing time): 424962 s

Lower Bound of LAC 1 (First Location Assignment Cost): 1 s

Lower Bound of LAC 2 (Second Location Assignment Cost): 0 UM (UM: Unproductive Moves).

Remark: We refuse any solution with makespan gap upper to 0.1.

In next Tables and figures we present the same elements as in the previous part.

Best and Worst Gap of approximated Pareto-Front

	C_{Max}	|V|	$\|ASC^*\|$	$\|P^*\|$	$\sum_{v\in ALV} t_v$	$\sum_{c\in ASC} t_c$	LAC_1	LAC_2
Best gap	0	0	0	0.225	0.018	0.05	10	0
Worst gap	0.081	0.32	15.09	8.009	0.383	21,2	40	400

Table 18

The objective values of solutions elected by indicators have objective values presented in next table:

Objective values of elected solutions

	IND	C_{Max}	|V|	|ASC*|	|P*|	$\sum_{v \in ALV} t_v$	$\sum_{c \in ASC} t_b$	LAC_1	LAC_2
Solution 1	349	28670	15	13	47	426907	53573	24	376
Solution 2	581	27052	16	12	138	429546	151950	18	380
Solution 3	334	27335	16	13	132	434135	39953	22	368
Solution 4	334	27335	16	13	132	434135	39953	22	368

Table 19

Objective gap of elected solutions

	IND	C_{Max}	|V|	|ASC*|	|P*|	$\sum_{v \in ALV} t_v$	$\sum_{c \in ASC} t_b$	LAC_1	LAC_2
Solution 1	349	0.06	0.062	0.181	0.324	0.05	0.44	23	376
Solution 2	581	0	0	0.09	0.24	0.06	3.089	17	380
Solution 3	334	0.01	0	0.181	0.37	0.07	0.07	21	368
Solution 4	334	0.01	0	0.181	0.37	0.07	0.07	21	368

Table 20

Solution 1: Solution proposed by first indicator.

Solution 2: Solution proposed by second indicator.

Solution 3: Solution proposed by third indicator.

Solution 4: Solution proposed by fourth indicator.

IND: Index of the solution in Pareto-List which represent the approximate Pareto-Front. All the elements of Pareto-List are presented is annexed parts.

We evaluate the gap of every objective value for each solution proposed by indicators of efficiency as described in part 3.7.2.1.

In table 14, we present the gap of each solution proposed by indicators for efficiency considering every objective.

Two indicators of efficiency propose the same solution, then we can elect that solution without taking into account the 2D-Projections of approximated Pareto Front. In next Figures we present 2D-Projections of Pareto Front approximation considering another resolution process of the same instances. Indicators of efficiency propose for that resolution distinct solutions.

2D-Projection of Pareto-Front Approximation

Plane of

Total ASC handling Time

AND

First Location Assignment Cost – LAC_1

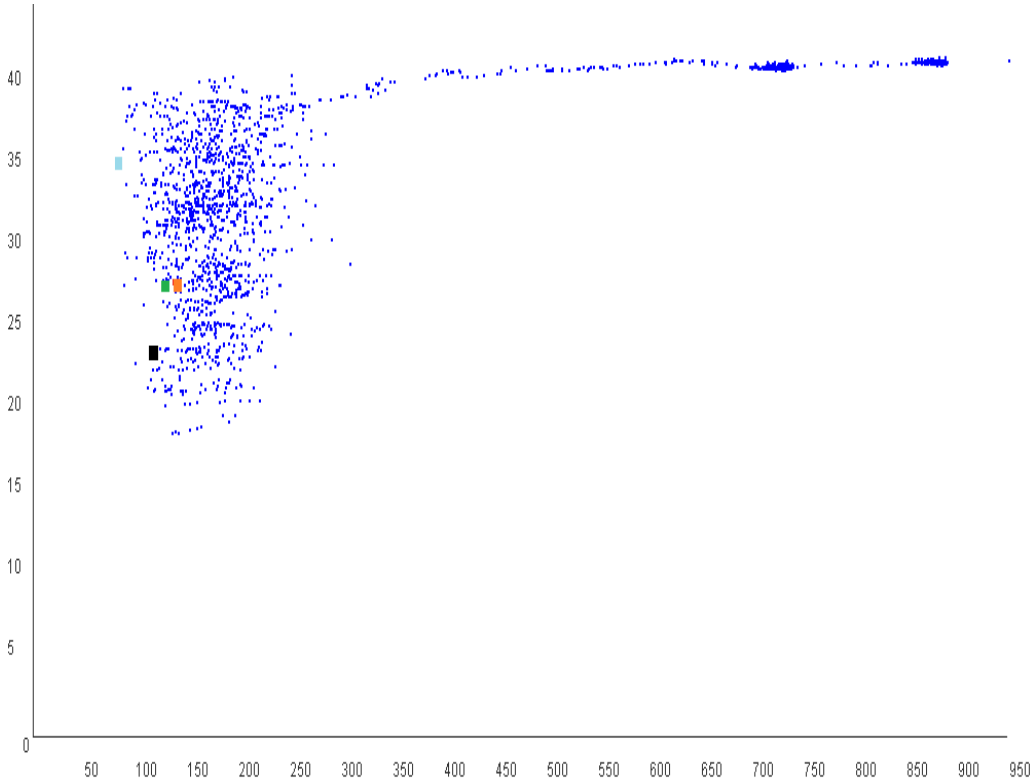


Figure 26

2D-Projection of Pareto-Front Approximation

Plane of

Total ALV Routing time

AND

First Location Assignment Cost – LAC_1

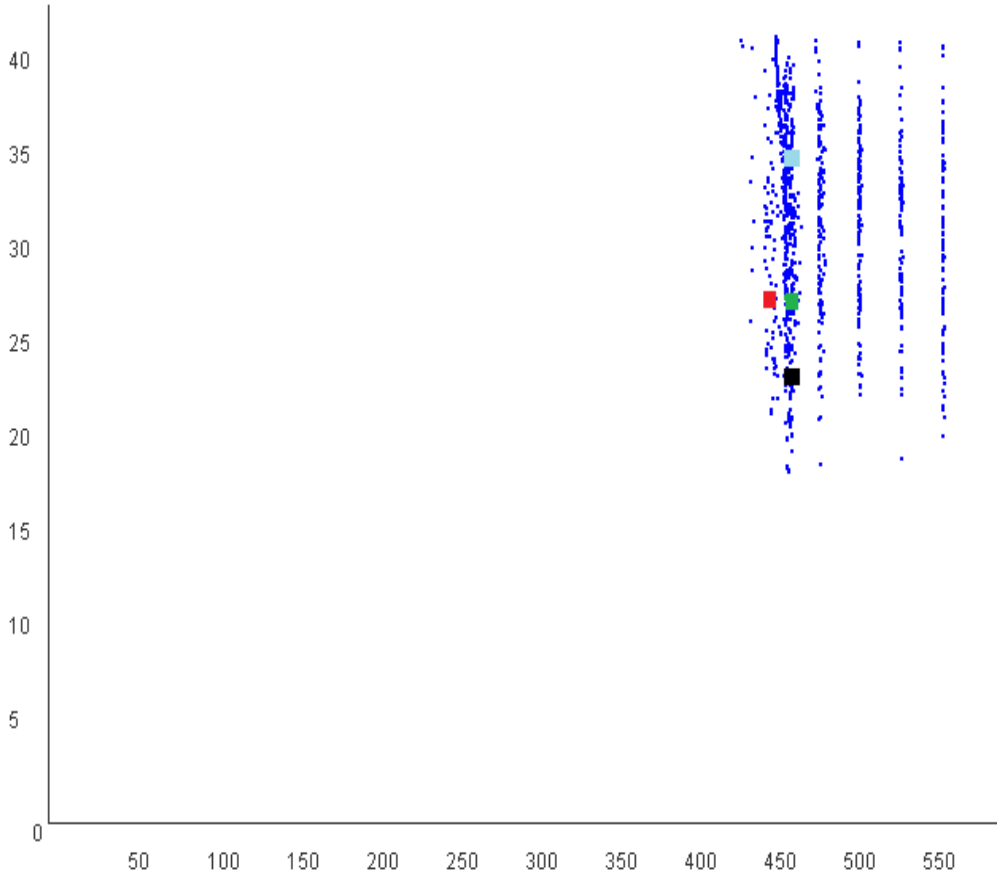


Figure 27

2D-Projection of Pareto-Front Approximation

Plane of

Second Location Assignment Cost - LAC_2

AND

First Location Assignment Cost – LAC_1

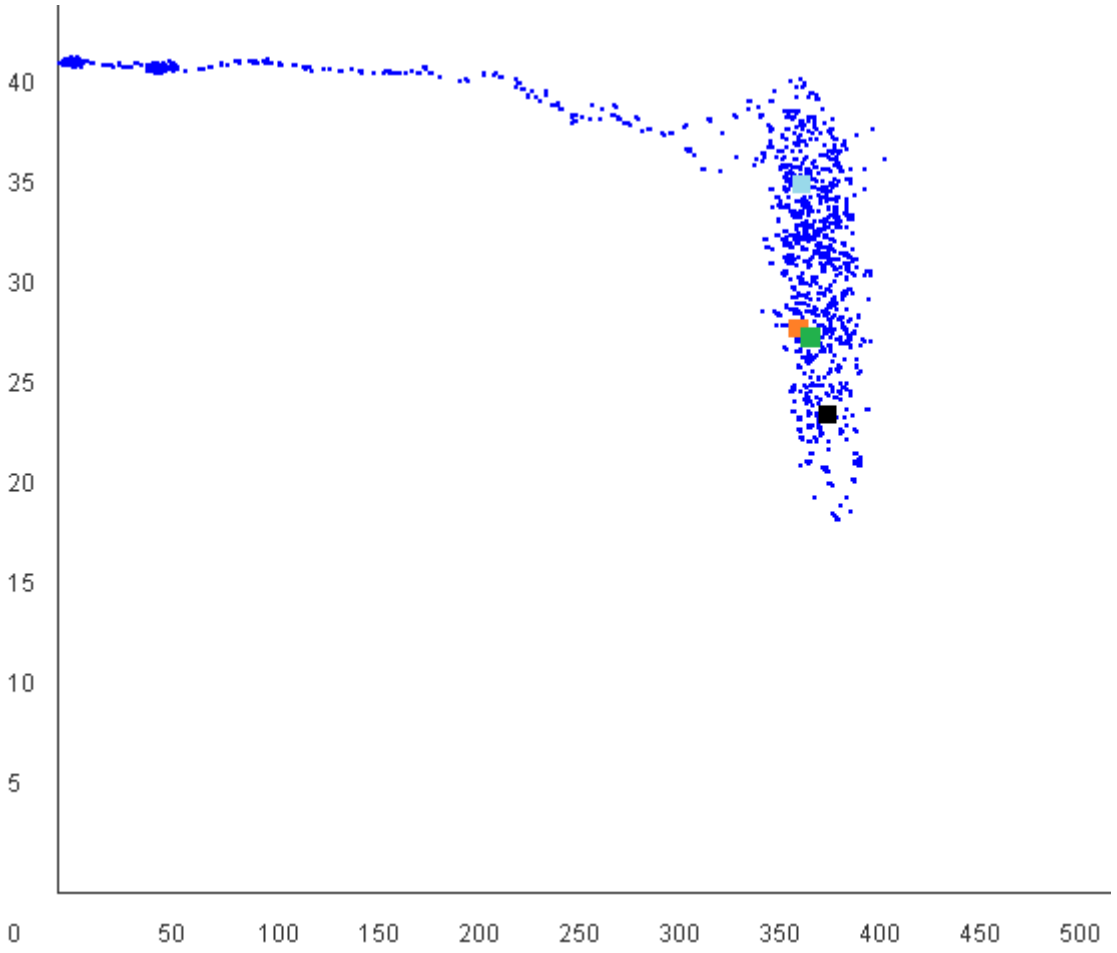


Figure 28

2D-Projection of Pareto-Front Approximation

Plane of

Number of bays to be used - |B*|

AND

Number of locations to be used – |P*|

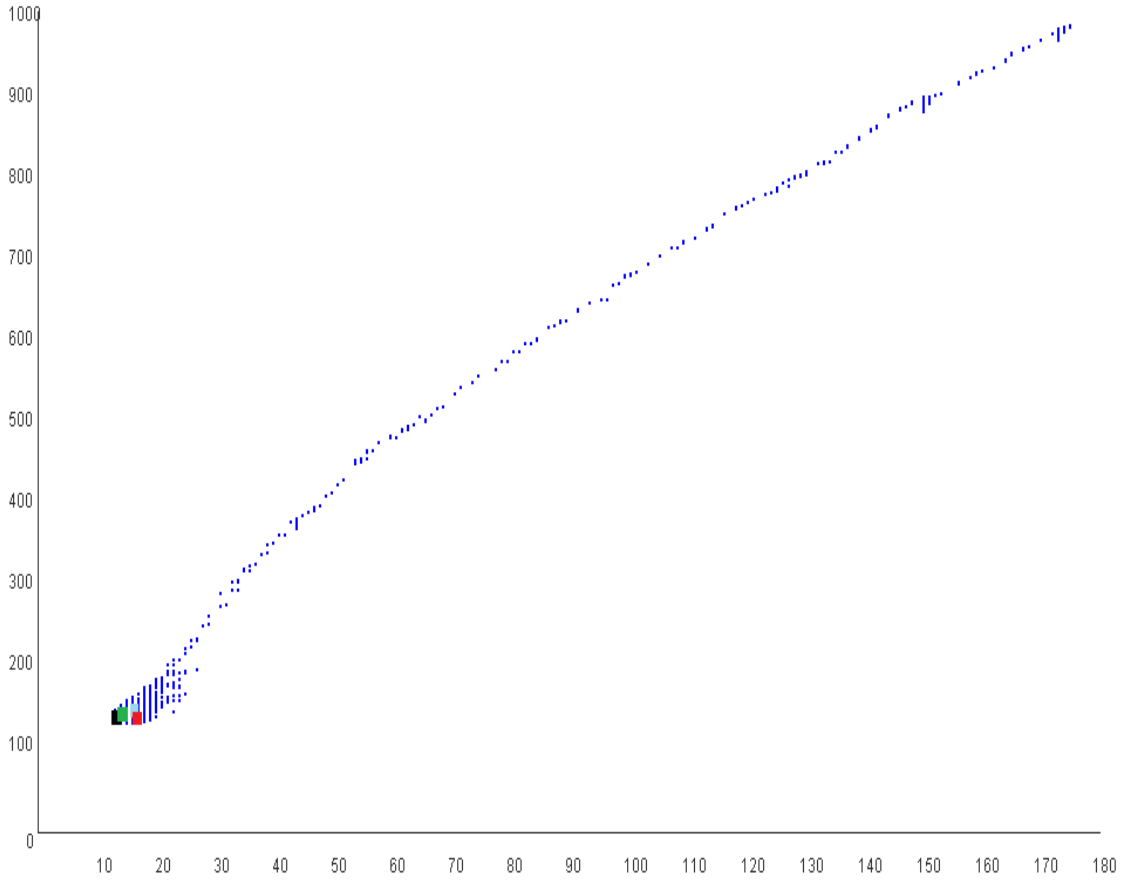


Figure 29

For the three case of MACT, if we consider the approximated Pareto-Front, the different lower bounds are reached efficiently seven objectives from eight. For the last objective, which is the first Location assignment cost and which represent the average distance between a container and the other containers to be delivered for the same client, short values of routing times between two containers of same client are reached (generally less than 10 seconds). Considering practice these values insure high quality of storage space.

3.8) Conclusion

In this part, we explore a problem which has not been extensively studied: the multi-objective integrated problem of location assignment and vehicle scheduling (IPLAVS) in maritime automated container terminal at import. As we know, this work is the second study of this specific problem considering container terminals and the first study of the problem considering straddle carrier terminals.

Considering our approach, in one hand, the integration of two optimization problems is theoretical guaranty of higher optimality. In the other hand, our solution proposes an 8-objective optimization process. It is also a new and efficient approach considering the real-world significance of the optimized objectives. We proved that the problem is NP-Complete using polynomial reductions of IPLASS to the parallel machine problem (PMP) and the Traveling Salesman Problem (TSP). We studied the numerical aspect of the total routing path variation considering a chosen layout. We developed a Multi-Objective Tabu Search Algorithm (MOTSA) adapted to the studied problem.

In the last part, we presented numerical results of MOTSA considering a large real instance of 1000 containers and a total storage space capacity of 10 000 containers. Four indicators of efficiency are evaluated for each solution of the approximated Pareto-Front in order to elect efficient solutions. The first and second indicators of efficiency are adapted to the combinatorial aspect of the problem, to its non-convexity, to the non-convexity of the Pareto Front Region. The third indicator is adapted to be easily used by operators in container terminal. The fourth indicator considers the two aspects. 2D-projections of the approximated Pareto Front are proposed to elect with efficiency one solution from the solutions proposed by indicators of efficiency. Considering the approximated Pareto-Front, the different lower bounds are reached efficiently considering the gap of every objective and realistic needs of operators in container terminal for large instance of 1000 containers 10 000 free storage locations.

Considering the approximated Pareto-Front, the different lower bounds are reached efficiently seven objectives from eight. For the last objective, which is the first Location assignment cost and which represent the average distance between a container and the other containers to be delivered for the same client, the value of 5.2 second of routing time between two containers of same client is reached. Considering practice these values insure high quality of storage space.

Conclusion

Maritime Container Terminals (MACT) plays a crucial role in global logistic networks. Because of the ever-increasing quantity of cargo, terminal operators need solutions for different decisional problems. In the maritime terminal, at boat arrival or departure, we observe five main problems: the allocation of berths, the allocation of query cranes, the allocation of storage space, the optimization of stacking cranes work load and the scheduling and routing of vehicles.

A good cooperation between the different installations in the terminal is important in order to optimize the productivity of Container Handling System (CHS). In an automated container terminal numerical solutions have become essential to optimize operators' decisions. Many recent researches have discussed the optimization of equipment scheduling in Maritime Automated Container Terminal (MACT).

We identify three kinds of MACT, considering their equipment; MACT using Automated Guided Vehicles (AGVs), Automated Stacking Cranes (ASCs) and Quay Cranes (QCs), MACT using ALVs, ASCs and QCs and MACT using Auto-Straddle-Carriers and QCs (without ASCs). In our study, we consider these three situations in Maritime Automated Container Terminal (MACT) at import.

In our study, we consider two optimization problems in automated container terminals at import; the first is the vehicle scheduling problem; the second is the integrated problem of location assignment and vehicle scheduling.

In the first part of our study, we propose different traffic layout adapted to the two studied problems and to every kind of automated container terminal. We present also relevant reviews of literature treating the optimization of container handling systems at maritime container terminal, the optimization of general automated guided vehicle system and the muti-objective optimization in general and in particular context of maritime container terminals.

In the second part, we resolve the planning of QC-AV-ASC. We present an effective model for every kind of traffic layout. We propose an efficient bi-objective model, which is important to determine the optimal storage time and the minimal number of AVs required. The bi-objective model can resolve large instances (until 500 containers) with double optimality (giving the optimal makespan and the minimum number of required AVs) in reasonable run time (less than 60 s). Our bi-objective model is perhaps the first model optimizing in on time the makespan and the AV fleet size in an automated container terminal. Our models consider 3 handling equipments (AV, QC and ASC) which is an efficient approach. We treated the three existing AVs: AGVs, ALVs and Auto-Strads.

In the third part of our work, we explore a problem which has not been extensively studied: the integrated problem of location assignment and vehicle scheduling (IPLAVS) in automated maritime container terminal at import. For resolution we considered two aspect of the problem: a mono-objective aspect and a multi-objective aspect. As we know, this work is the second study of this specific problem considering container terminals and the first study of the problem considering multi-objective aspect. For mono-objective IPLAVS, we developed a new cooperative tabu search and we prove its efficiency for solving combinatorial problems using an application to the Traveling Salesman Problem.

Considering our approach, in one hand, the integration of two optimization problems is theoretical guaranty of higher optimality. In the other hand, our solution proposes an 8-objective optimization process. It is also a new and efficient approach considering the real-world significance of the optimized objectives. We proved that the problem is NP-Complete using polynomial reductions of IPLASS to the parallel machine problem (PMP) and the Traveling Salesman Problem (TSP). We studied the numerical aspect of the total routing path variation considering a chosen layout. We developed a Multi-Objective Tabu Search Algorithm (MOTSA) adapted to the studied problem. In the last part, we presented numerical results of MOTSA considering a large real instance of 1000 containers and a total storage space capacity of 10 000 containers. Four indicators of efficiency are evaluated for each solution of the approximated Pareto-Front in order to elect efficient solutions. The first and second indicators of efficiency are adapted to the combinatorial

aspect of the problem, to its non-convexity, to the non-convexity of the Pareto Front Region. The third indicator is adapted to be easily used by operators in container terminal. The fourth indicator considers the two aspects. 2D-projection of the approximated Pareto Front are proposed to elect with efficiency one solution from the solutions proposed by indicators of efficiency. Considering the approximated Pareto-Front, the different lower bounds are reached efficiently considering the gap of every objective and realistic needs of operators in container terminal for large instance of 1000 containers 10 000 free storage locations.

Different parts of our study was published and communicated in international conferences.

Publication in book chapter

1. H. DKHIL, A. YASSINE, H. CHABCHOUB: Optimization of Container Handling Systems in Automated Maritime Terminal. Advanced Methods for Computational Collective Intelligence Studies in Computational Intelligence (Springer), Vol. 457, pp. 301-312, 2013. http://link.springer.com/chapter/10.1007%2F978-3-642-34300-1_29

Publications in international journals

1. Multi-objective optimization of the integrated problem of location assignment and straddle carrier scheduling in maritime container terminal at import. Accepted for publication in Journal of Operational Research Society
2. A new collaborative meta-heuristic approach: application to the Traveling Salesman Problem, Accepted for publication in International Journal of Artificial Intelligence

International conferences

1. H. DKHIL, A. YASSINE, H. CHABCHOUB: Optimization of Container Handling Systems in Automated Maritime Terminal. International Conference on Computational Collective Intelligence - Technologies and Applications (ICCCI 2012), Ho Chi Minh City –Vietnam, 28-30 novembre 2012.
2. H. DKHIL, A. YASSINE, H. CHABCHOUB: The AGV scheduling problem: Metaheuristic approach with genetic and hybrid algorithms. International Conference on Metaheuristics and Nature Inspired Computing (Meta 2012), Sousse – Tunisie, 27-31 octobre 2012.

3. H. DKHIL, A. YASSINE, H. CHABCHOUB: Optimisation bi-objective du problème de stockage de conteneurs dans un terminal maritime. International Conference on Logistics Operations Management (GOL'12), Université du Havre - France, 17-19 octobre 2012.

http://marlog-aast.org/2013/Papers/S3P1.pdf

4. H. DKHIL, A. YASSINE, H. CHABCHOUB: Optimization of container handling systems in automated maritime terminals: A hybrid genetic and a tabu search algorithms. International Conference on Advanced Logistics and Transport (ICALT 2013), pp. 539-544, Sousse - Tunisia, 29-31 Mai 2013.

http://ieeexplore.ieee.org/xpl/articleDetails.jsp?reload=true&tp=&arnumber=6568516

5. H. DKHIL, A. YASSINE, H. CHABCHOUB: TITRE 4. European Conference on Operational Research (Euro Informs 2013), Roma – Italy, 1-4 Juliet 2013.

6. H. DKHIL, A. YASSINE, H. CHABCHOUB: TITRE 2. International Conference on Metaheuristics and Nature Inspired Computing (Meta 2014), Marrakech – Morocco, 27-31 octobre 2014.

7. H. DKHIL, A. YASSINE, H. CHABCHOUB: Optimization and Simulation of Container Handling Systems in Automated Maritime Terminal. THE INTERNATIONAL MARITIME TRANSPORT & LOGISTICS CONFERENCE (MARLOG 2): Sustainable Development of Suez Canal Region, Alexandria – Egypt, 17-19 mars 2013.

In next works, we will study the dynamic integrated problem of location assignment and vehicle scheduling in maritime automated container terminals at import. We will study also the export case and the general case of import-export with dynamic aspect.

References

[1] Meersmans, P.J. M. (2002), Optimization of container handling systems, Ph.D. Thesis, Tinbergen Instutue 271 Erasmus University, Rotterdam.

[2] Muler, T. (1983), Automated Guided Vehicles, IFS (Publications) Ltd, UK, Springer-Verlag, Berlin.

[3] Maxwell, W.L., Muckstadt, J.A. (1982), Design of automatic guided vehicle systems, IIE Transactions 14(2), 114-124.

[4] Rajotia, S., Shanker, K., Batra, J.L (1988), Determination of optimal AGV fleet size for an FMS, International Journal of Production Research 36(5), 1177-1198.

[5] Sinriech, D., Tanchoco, J.M.A. (1992), An economic model determining AGV fleet size, International Journal of Production Research 30(6), 1255-1268.

[6] IFA Vis, R de Koster, KJ Roodbergen, LWP Peeters (2001), Determination of the number of automated guided vehicles required at a semi-automated container terminal, 52, 409-417.

[7] Phillips, D.T., Garsia-Diaz, A. (1981), Fundamentals of network Analysis, Prentice Hall, Inc., Englewood Cliffs, New York.

[8] Ford, L.R., Fulkerson, D.R. (1962), Flows in Networks, Princeton University Press, Princeton, New Jerey.

[9] Dantizing, G.B., Fulkerson, DR. (1945), Minimizing time number of tankers to meet a fixed schedule, Naval Research Logistics Quaterly 1, 217-222.

[10] Ahuja, R.K, Magnati, T.L., Orlin, J.B. (1993), Network Flows, Theory, Algorithms, and Appliquations, Prentice Hall, New Jersy.

[11] Egbelu, P.J. and Tanchoco, J.M.A. (1984). Characterization of automated guided

vehicle dispatching rules. International Journal of Production Research, 22:359-374.

[12] Umit Bilge, J.M.A. Tanchoco, (1997), AGV Systems with Multi-Load Carriers: Basic Issues and Potential Benefits, Journal of Manufacturing Systems,16.

[13] Chen, Y., Leong, T-Y., Ng, J.W.C., Demir, E.K., Nelson, B.L., and Simchi-Levi, D. (1997).Dispatching automated guided vehicles in a mega container terminal. The National University of Singapore/Dept. of IE & MS, Northwestern University.

[14] Kim, K.H. and Bae, J.(1999). A dispatching method for automated guided vehicles to minimize delays of containership operations. International Journal of Management Science, 5:1-25.

[15]Bae M-K, Park Y-M, Kim KH (2007) A dynamic berth scheduling method. Paper presented at the international. conference on intelligent manufacturing and logistics systems (IML 2007), Kitakyushu, Japan, 26–28 February 2007.

[16] Kim, K.H. (1997), Evaluation of the number of re-handles in container yards, Computers & Industrial Engineering, 32 (4), 701-711.

[17] Kim, K.H. Kim, K.Y. (1999), Routing straddle carriers for the loading operation of containers using a beam search algorithm, Computers & Industrial Engineering, 36 (1), 109-136.

[18] Kim, H.K. Kim, H.B. (1999), Segregating space allocation models for containers inventories in port container terminals, International Journal of Production Economics, 59 (1-3), 415-423.

[19] Kim, K. Park, Y. Ryu, K. (2000), Driving decision rules to locate export containers in container yards, European Journal of Operational Research, 124 (1), 89-101

[20] Chen, L. Bostel, N. Dejax, P. Cai, J. Xi, L. (2007), A tabu search algorithm for the integrated scheduling problem of container handling systems in a maritime terminal, European Journal of Operational Research,181, 40-58

[21] Dahlstrom , K.J and Maskin, T., 1981, Where to use AGV systems, manual fork lifts, traditional fixed roller conveyor systems

respectively. Proceedings of the 1st International AGV Conference, 173-182.

[22] Muller, T., Comparaison of operating costs between different transportation systems. Proceedings of the 1st International AGV Conference, 145-155.

[23] Sinirech, D., Tanchoco, JMA (1992), AN economic model for determining AGV fleet size, International Journal of Production Research 29(9), 1725-1268.

[24]Vu Duc Ngyuyen, Kap Hwan Kim (2009), A dispatching method for automated lifting vehicles in automated port container terminals. Computers and industrial engineering, 56, 1002-1020.

[25] Maxwell, W.L., Muckstadt, J.A., Design of automatic guided vehicle systems, IIE Transactions, 14(2), 1982, pp. 114-124.

[26] Rajotia, S., Shanker, K., Batra, J.L, Determination of optimal AGV fleet size for an FMS, International Journal of Production Research, 36(5), 1988,pp. 1177-1198.

[27] Golias, M.N. Theofanis, S. Boile, M. (2009), A bi-level formulation of the berth scheduling problem with variable vessel release dates to reduce port emissions, Proceeding of 2009 International Conference on Shipping, Ports, Logistics Management, Inha University, korea, ISLC, Incheon.

[28] Giallombardo, G. Moccia, L. Salani, M. Vacca, I (2010).The tactical berth allocation problem with quay crane assignment and transshipment-related quadratic yard costs.In: Proceedings of the EuropeanTransport Confeence (ETC),1–27.

[29] Bish, E.K. Leong, T. Li, C. Ng, J.W.C. Simchi-Levi, D. (2001), Analysis of a new Vehicle Scheduling and Location Problem, Naval Research Logistics 48, 363-385.

[30] Deb, K. Pratap, A. Agarwal, S. Meyarivan, T. (2002), A fast and elitist multi-objective genetic algorithm: NSGA-II, Evolutionary Computation, IEEE Transactions 6 (2), 182-197.

[31] Jaeggi, D.M. Parks, G.T. Kipouros, Clarkson, P.J. (2006), The development of multi-objective Tabu Search algorithm for continuous optimization problems, European Journal of Operational Research,185 (3), 1192-1212.

[32] Hansen M. (1997), Tabu Search for multi-objective optimization: MOTS. In: MCDM Cape Town, South Africa.

[33] Dawson, P. Jaeggy, D. Parks, G. Molina-Cristobal, A. Clarkson, P.J. (2007), The Development of Multi-thred Multi-objective Tabu Search Algorithm, Evolutionary Multi-Criterion Optimization, Lecture Notes in Computer Science, 4403, 242-256.

[34] Jaegyy, D. Asselin-Miller, C. Parks, G. Kipouros, T. Bell, T. Clarkson, J. (2004), Multi-objective Parallel Tabu Search, Parallel Problem Solving From Nature, Lecture Notes in Computer Science, 3242, 732-741.

[35] Tuyttens, D. Teghem, J. Fortemps, Ph. Van Nieuwenhuyze, K. (2000), Performance of the MOSA method for the bi-criteria assignment problem, Journal of Heuristics, 6, 295-310.

[36] Loukil, T. Teghem, J. Tuyttens, D. (2005), Solving multi-objective production scheduling problems using meta-heuristics, European Journal of Operational Research, 161, 42-61.

[37] Ulungu, E.L. Teghem, J. Fortemps, P.H. (1999), MOSA method, a tool for solving multi-objective combinatorial optimization problems, Journal of Multi-Criteria Decision Analysis, 8, 221-236.

[38] Gandibleux, X.Feville, A. (2000), Tabu Search Based Procedure for solving the 0/1 multi-objective knapsack problem: the two objective case. Journal of Heuristics, 6 (3), 361-383.

[39] Hansen, M.P. (2000) Tabu Search for multi-objective combinatorial optimization: Tamoco, Control and Cybernetics, 29 (3), 799-818.

[40] Chang Ho Yang and all.

[41] Glover, F. ; Laguna, M. (1997),” Tabu search”, Kluwer Academic Publishers, Dordrecht.

[42] M. El-Abd and M. Kamel. (2005) , “A taxonomy of cooperative search algorithms,” LNC3 3636, 32–41

[43] T.G. Crainic, M. Toulouse and M. Gendreau. (1997), “ Towards a Taxonomy of Parallel tabu search heuristics”, INFORMS Journal on computing, vol 9, no .1, 61 –72.

[44] M. Nowostawski and R. Poli. (1999), “ Parallel genetic algorithm taxonomy”, 3 rd international conference on knowledge-based intelligent information Engineering systems, 88– 92.

[45] D.R. Greening, (1990) ”Parallel Simulated Annealing Techniques”. Physica D, vol. 42, 293-306.

[46] E. K. Burke and Y. Bykov. (2008) "A Late Acceptance Strategy in Hill-Climbing for Exam Timetabling Problems", (extended abstract), in proccedings of PATAT 2008 conference. Montreal, Canada, August 2008.

[47] E. K. Burke and Y. Bykov, (2012) "The Late Acceptance Hill-Climbing Heuristic", Technical Report CSM-192, Computing Science and Mathematics, University of Stirling.